

# PNAS



1

## 2 **Supporting Information for**

### 3 **Joint Trajectory Inference for Single-cell Genomics Using Deep Learning with a Mixture Prior**

4 **Jin-Hong Du, Tianyu Chen, Ming Gao, Jingshu Wang**

5 **Jingshu Wang.**

6 **E-mail: [jingshuw@uchicago.edu](mailto:jingshuw@uchicago.edu)**

#### 7 **This PDF file includes:**

8 Supporting text

9 Figs. S1 to S8

10 Tables S1 to S2

11 SI References

## 12 Supporting Information Text

13 Below, we outline the structure of the rest of the supplement. Section S1 introduces the proposed trajectory inference method,  
14 while the specifications of certain algorithm details are deferred to Section S2. The details of simulation datasets, evaluation  
15 metrics, and experiment procedures are presented in Section S3. The details of case studies on mouse brain datasets are  
16 provided in Section S4. The sensitivity of hyperparameters and computational efficiency are analyzed in Section S5. Lastly, the  
17 supplementary tables and extra experimental results on real data are included in Section S6 and Section S7, respectively.

## 18 Contents

19	<b>S1 Methods</b>	<b>2</b>
20	S1.1 Model	2
21	S1.2 Trajectory Inference from Posterior Approximations	5
22	S1.3 Differential gene expression along the trajectory	6
23	S1.4 Model estimation with practical considerations	7
24	S1.5 Discussion on Uncertainty Quantification	7
25	<b>S2 Technical details</b>	<b>8</b>
26	S2.1 Loss Function	8
27	S2.2 Posterior Estimation	10
28	S2.3 Edge Score	11
29	S2.4 Proof of Proposition 1	12
30	<b>S3 Benchmarking</b>	<b>14</b>
31	S3.1 Benchmarking Datasets	14
32	S3.2 Evaluation metrics	14
33	S3.3 DE analysis	15
34	<b>S4 Real data</b>	<b>15</b>
35	S4.1 Experiments on mouse brain datasets	15
36	S4.2 Experiments on the integration of three mouse brain datasets	15
37	S4.3 Experiments on multi-omic human hematopoiesis datasets	16
38	<b>S5 Analysis of hyperparameters sensitivity and computational efficiency</b>	<b>16</b>
39	S5.1 Hyperparameter $\gamma$ for Jacobian regularization	16
40	S5.2 Hyperparameter $\alpha$ for the Soft penalty term	16
41	S5.3 Hyperparameter $\kappa$ for MMD regularization	17
42	S5.4 Computation efficiency and memory usage	17
43	<b>S6 Supplementary tables</b>	<b>19</b>
44	<b>S7 Supplementary figures</b>	<b>20</b>

## 45 S1. Methods

### 46 S1.1. Model.

47 **A hierarchical mixture model for the trajectory structure** Inspired by the common trajectory model proposed in (1), we start with  
48 the trajectory backbone defined on a complete graph  $\mathcal{G} = (\mathcal{N}, \mathcal{E})$  with vertices  $\mathcal{N}(\mathcal{G})$  and edges  $\mathcal{E}(\mathcal{G})$ . Let the  $k$  vertices in  
49  $\mathcal{N}(\mathcal{G})$  be the distinct cell states. We use an edge between two vertices to represent the transitioning between two states. To  
50 model the scenario that one cell either belongs to a specific state or is developing from one to another, we assume that each cell  
51 is positioned either on one vertex or an edge. Specifically, let  $\tilde{w}_i \in [0, 1]^k$  be the position of cell  $i$  on the backbone, we have

$$52 \quad \tilde{w}_i = \begin{cases} e_j & \text{if cell } i \text{ is on vertex } j \in \{1, \dots, k\} \\ w_i e_{j_1} + (1 - w_i) e_{j_2} & \text{if cell } i \text{ is on the edge between vertices } j_1 \text{ and } j_2 \text{ } (j_1 \neq j_2) \end{cases}$$

53 where  $e_j$  is a one-hot vector with  $j$ th element 1 and all other elements 0, and  $w_i \in [0, 1]$  describes the relative position of cell  $i$   
54 if it is on an edge.

55 The basic goal of our trajectory inference is to infer the trajectory backbone  $\mathcal{B}$  - a subgraph of  $\mathcal{G}$  - whose edges have positive  
56 proportions of cells:

$$57 \quad \mathcal{N}(\mathcal{B}) = \mathcal{N}(\mathcal{G})$$
$$58 \quad \mathcal{E}(\mathcal{B}) = \left\{ (j_1, j_2) \in \mathcal{E}(\mathcal{G}) : \sum_i \mathbb{1}_{\{\tilde{w}_{ij_1} > 0, \tilde{w}_{ij_2} > 0\}} > 0 \right\}.$$

59 Meanwhile, we also aim to estimate the relative positions  $\tilde{\mathbf{w}}_i$  of each cell on this trajectory backbone and, as in other TI  
60 methods, the pseudotime order of the cells along the trajectory. Intuitively, the pseudotime corresponds to the biological  
61 progression of a cell through the dynamic process that results in the trajectory structure (2).

62 Here, we give a formal definition of the pseudotime in our framework. Assume that each edge  $\ell \in \mathcal{E}(\mathcal{B})$  is associated with  
63 a duration (an edge weight)  $b_\ell$ . Without external information, we set  $b_\ell = 1$ . Given a root vertex  $k_0$ , we first generate a  
64 directed trajectory backbone  $\vec{\mathcal{B}}$ , by giving each edge in  $\mathcal{E}(\mathcal{B})$  a direction. We start with finding the connected components  
65  $\mathcal{A} \subset \mathcal{N}(\mathcal{B})$  containing  $k_0$  on the undirected trajectory backbone  $\mathcal{B}$ . Any edges that do not connect to a node in  $\mathcal{A}$  become a  
66 bi-directional edge in  $\vec{\mathcal{B}}$ . If the subgraph of  $\mathcal{B}$  with nodes in  $\mathcal{A}$  forms a tree, then given the root  $k_0$ , we turn it into a directed  
67 tree starting from  $k_0$ . However, if the undirected subgraph of  $\mathcal{B}$  with nodes in  $\mathcal{A}$  contains any cycles, we can not automatically  
68 determine the directions of all edges on the subgraph given the root. Our current implementation has two options. By default,  
69 we automatically find the minimum spanning tree of the subgraph of  $\mathcal{B}$  on  $\mathcal{A}$  into a tree by finding the minimum spanning tree  
70 of the subgraph and then subsequentially turn it into a directed tree. Alternatively, we ask the user to specify the direction of  
71 every edge through a `networkx.DiGraph` object.

72 The pseudotime for vertex  $j$  is defined as:

$$73 \quad o_j = \begin{cases} \max \left\{ \sum_{\ell \in \mathcal{L}} b_\ell : \mathcal{L} \subset \mathcal{E}(\vec{\mathcal{B}}) \text{ forms a simple path from } k_0 \text{ to } j \right\} \\ \infty \quad \text{if no such } \mathcal{L} \text{ exists} \end{cases} \quad [1]$$

74 where a simple path is a path that does not have any repeating vertices. Let  $\mathbf{o} = (o_1, \dots, o_k)$ , then the pseudotime of the cell  $i$   
75 is defined as

$$76 \quad T_i = \mathbf{o}^\top \tilde{\mathbf{w}}_i \quad [2]$$

77 which equals the pseudotime of the vertex that the cell is on or the weighted average of the pseudotime of the two vertices of  
78 the edge that the cell belongs to.

79 By our definition, any vertex that precedes another vertex on a simple path that starts from  $k_0$  has a smaller pseudotime  
80 than the other vertex. Thus our definition guarantees a meaningful ordering of the cells when the directed trajectory backbone  
81  $\vec{\mathcal{B}}$  does not contain any cycles. If  $\vec{\mathcal{B}}$  contains cycles, the pseudotime of the cells remains well-defined under our definition but  
82 may not be biologically meaningful.

83 Now, we relate the observed single-cell sequencing data with the underlying trajectory backbone graph  $\mathcal{B}$ . Let  $\mathbf{Y}_i =$   
84  $(y_{i1}, \dots, y_{iG})$  be the observed counts of  $G$  features in cell  $i$ . Though  $\mathbf{Y}_i$  is the vector of observed counts with complicated  
85 dependence across features, we assume that the dependence can be explained by latent Gaussian variables  $\mathbf{Z}_i \in \mathbb{R}^d$ , which is in  
86 a space with a much lower dimension and associated with the trajectory backbone  $\mathcal{B}$ . We also take into account the effects of  
87 known confounding covariates  $\mathbf{X}_i \in \mathbb{R}^s$  (such as cell-cycle or batch effects, which we take as deterministic variables) on  $\mathbf{Y}_i$ .  
88 Specifically, we assume the following latent variable model on the observed counts:

$$89 \quad \begin{aligned} \mathbf{Z}_i | \tilde{\mathbf{w}}_i &\sim \mathcal{N}_d(\mathbf{U} \tilde{\mathbf{w}}_i, \mathbf{I}_d) \\ Y_{ig} | \mathbf{Z}_i, \mathbf{X}_i &\stackrel{\text{ind}}{\sim} \mathcal{NB}(l_i f_g(\mathbf{Z}_i, \mathbf{X}_i), \theta_g), \quad g = 1, 2, \dots, G \end{aligned} \quad [3]$$

90 Here  $\mathbf{U} \in \mathbb{R}^{d \times k}$  is the unknown matrix of the vertices positions which, together with the cells' relative positions on  $\mathcal{B}$ , determines  
91 the means of  $\mathbf{Z}_i$ . Then, we assume that the observed counts  $\mathbf{Y}_i$  nonlinearly depend on the latent variables  $\mathbf{Z}_i$ . The scalar  
92  $l_i$  is the known library size of cell  $i$ , and each  $f_g : \mathbb{R}^{d+s} \rightarrow \mathbb{R}, g = 1, 2, \dots, G$  is an unknown nonlinear function involving  
93 the confounding covariate vector  $\mathbf{X}_i$ . The unknown parameters  $\theta_g$  are feature-specific dispersion parameters of the Negative  
94 Binomial (NB) distribution. Notice that though the edges are assumed to be linear lines in the latent space, they are likely  
95 curves in the observed data space via the nonlinear mappings  $\{f_g(\cdot), g = 1, 2, \dots, G\}$ .

96 The assumption of using NB distributions to model single-cell sequencing data is based on a detailed review and discussion  
97 in (3). Specifically, for scRNA-seq with the unique molecular identifier (UMI) counts, NB distributions can describe the  
98 stochasticity in scRNA-seq, accounting for both biological and technical noise. However, they may not be adequate for  
99 sequencing data where the non-zero counts are large, such as scRNA-seq data without UMI. In that scenario, we assume that  
100 the observed counts follow zero-inflated Negative Binomial (ZINB) distributions, as in (4):

$$101 \quad Y_{ig} | \mathbf{Z}_i, \mathbf{X}_i \stackrel{\text{ind}}{\sim} \phi_{ig} \delta_0 + (1 - \phi_{ig}) \mathcal{NB}(l_i f_g(\mathbf{Z}_i, \mathbf{X}_i), \theta_g) \quad [4]$$

102 where  $\phi_{ig} = h_g(\mathbf{Z}_i, \mathbf{X}_i)$  is the zero inflation probability with  $h_g(\cdot)$  an unknown non-linear function for  $g = 1, \dots, G$ .

103 The above model defines the trajectory backbone  $\mathcal{B}$ , cell positions  $\tilde{\mathbf{w}}_i$ , and pseudotime  $T_i$  that are identifiable from single-cell  
104 sequencing data. However, estimating these quantities is still challenging as the vertices and edges are defined on a latent space.  
105 Thus, we further impose a hierarchical model on  $\tilde{\mathbf{w}}_i$  to simplify model estimation. First, we introduce a latent categorical  
106 variable  $c_i$  as the index of all edges and vertices. Specifically, let  $c_i$  take values in  $\{1, 2, \dots, K\}$  where  $K = k(k+1)/2$  is the

number of all possible edges and vertices in  $\mathcal{G}$ . For the complete graph  $\mathcal{G}$ , define the categorical assignment symmetric matrix,

$$C = \begin{matrix} & 1 & 2 & \cdots & k \\ \begin{matrix} 1 \\ 2 \\ \vdots \\ k \end{matrix} & \begin{pmatrix} 1 & 2 & \cdots & k \\ 2 & k+1 & \cdots & 2k-1 \\ \vdots & \vdots & \ddots & \vdots \\ k & 2k-1 & \cdots & K \end{pmatrix} \end{matrix}. \quad [5]$$

When  $c_i$  equals to  $C_{j_1 j_2}$  for  $j_1 \neq j_2$ , the cell  $i$  is at the edge between vertex  $j_1$  and vertex  $j_2$ . When  $c_i$  equals to  $C_{jj}$ , the cell  $i$  is at the vertex  $j$ . We let  $C_{j_2 j_1} = C_{j_1 j_2} = j_1 + (j_2 - 1)k$  for  $j_1 \leq j_2$  and set  $c_i = C_{j_1 j_2}$  if cell  $i$  is at the edge between vertex  $j_1$  and vertex  $j_2$ , or if cell  $i$  is at the vertex  $j_1$  when  $j_2 = j_1$ . Then we assume the following mixture prior on  $\tilde{\mathbf{w}}_i$ :

$$\begin{aligned} w_i &\stackrel{\text{i.i.d.}}{\sim} \text{Uniform}(0, 1) \\ c_i &\stackrel{\text{i.i.d.}}{\sim} \text{Multinomial}(1, \boldsymbol{\pi}), \quad \boldsymbol{\pi} \in [0, 1]^K \\ w_i &\perp\!\!\!\perp c_i \\ \tilde{\mathbf{w}}_i &= w_i \mathbf{a}_{c_i} + (1 - w_i) \mathbf{b}_{c_i} \end{aligned} \quad [6]$$

where  $\mathbf{a}_c = \mathbf{e}_{j_1}$  and  $\mathbf{b}_c = \mathbf{e}_{j_2}$  if  $c = C_{j_1 j_2}$  for  $j_1 \leq j_2$ .

To summarize, combining models Eq. (3) and Eq. (6), we obtain a hierarchical mixture model with an underlying trajectory structure. The unknown parameters are the mean positions of the  $k$  cell types in the latent space  $\mathcal{U}$ , the prior probabilities of the  $K$  categories  $\boldsymbol{\pi}$ , the mapping functions  $f_g(\cdot)$ ,  $g = 1, 2, \dots, G$  and the dispersion parameters  $\theta_g$  (and also  $h_g(\cdot)$  for the ZINB model).

**VAE for approximating the posterior distributions** To introduce a wide class of non-linear mapping functions, we model  $f_g(\cdot)$  by a neural network and further combine our model with the variational autoencoder (VAE) (5) for approximating the posterior distributions. Following the variational Bayes approach in VAE and the conditional VAE (6), we use Gaussian distributions to approximate the intractable posterior distribution of  $\mathbf{Z}_i$  given the observed data  $\mathbf{Y}_i$  and confounding variables  $\mathbf{X}_i$ :

$$\mathbf{Z}_i | \mathbf{Y}_i, \mathbf{X}_i \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{Y}_i, \mathbf{X}_i}, \boldsymbol{\Sigma}_{\mathbf{Y}_i, \mathbf{X}_i}). \quad [7]$$

Here the ‘‘posterior’’ mean  $\boldsymbol{\mu}_{\mathbf{Y}_i, \mathbf{X}_i}$  and covariance  $\boldsymbol{\Sigma}_{\mathbf{Y}_i, \mathbf{X}_i} = \text{diag}(\sigma_{\mathbf{Y}_i, \mathbf{X}_i, 1}^2, \dots, \sigma_{\mathbf{Y}_i, \mathbf{X}_i, d}^2)$  are functions of  $(\mathbf{Y}_i, \mathbf{X}_i)$ . To guarantee flexibility, they are also modeled by a neural network. The neural network for  $\boldsymbol{\mu}_{\mathbf{Y}_i, \mathbf{X}_i}$  and  $\boldsymbol{\Sigma}_{\mathbf{Y}_i, \mathbf{X}_i}$  is the encoder, and the network for  $f_g(\mathbf{Z}_i, \mathbf{X}_i)$  is the decoder. For non-UMI data, as in (4), the functions  $h_g(\cdot)$  for zero-inflation parameters are also modeled in the decoder and share the same hidden layers as  $f_g(\cdot)$ .

With the above setup, we can lower bound the log-likelihood of each observation  $\mathbf{Y}_i$  as

$$\begin{aligned} \log p(\mathbf{Y}_i | \mathbf{X}_i) &= \log [\mathbb{E}_{p(\mathbf{Z}_i | \mathbf{X}_i)} p(\mathbf{Y}_i | \mathbf{Z}_i, \mathbf{X}_i)] \\ &\geq \mathbb{E}_{q(\mathbf{Z}_i | \mathbf{Y}_i, \mathbf{X}_i)} \log p(\mathbf{Y}_i | \mathbf{Z}_i, \mathbf{X}_i) - D_{\text{KL}}(q(\mathbf{Z}_i | \mathbf{Y}_i, \mathbf{X}_i) \| p(\mathbf{Z}_i)), \end{aligned} \quad [8]$$

where  $p(\mathbf{Y}_i | \mathbf{Z}_i, \mathbf{X}_i)$  is the true conditional distribution in model Eq. (3) or Eq. (4), and  $q(\mathbf{Z}_i | \mathbf{Y}_i, \mathbf{X}_i)$  is the Gaussian approximation of the posterior distribution in Eq. (7). This lower bound is often referred to as the evidence lower bound (ELBO) (5), where the first term  $\mathbb{E}_{q(\mathbf{Z}_i | \mathbf{Y}_i, \mathbf{X}_i)} \log p(\mathbf{Y}_i | \mathbf{Z}_i, \mathbf{X}_i)$  denotes the reconstruction likelihood and the second term  $-D_{\text{KL}}(q(\mathbf{Z}_i | \mathbf{Y}_i, \mathbf{X}_i) \| p(\mathbf{Z}_i))$  behaves as a regularizer. The difference between Eq. (8) and regular conditional VAE is that we have a mixture model for the trajectory structure encoded in the term  $p(\mathbf{Z}_i)$ , which encourages the posteriors of  $\mathbf{Z}_i$  to lie along linear edges and vertices. The resulting loss function for one cell is defined as the negative modified ELBOs in (8),

$$\mathcal{L}(\mathbf{Y}_i; \mathbf{X}_i, \Theta) = -\mathbb{E}_{q(\mathbf{Z}_i | \mathbf{Y}_i, \mathbf{X}_i)} \log p(\mathbf{Y}_i | \mathbf{Z}_i, \mathbf{X}_i) + \beta D_{\text{KL}}(q(\mathbf{Z}_i | \mathbf{Y}_i, \mathbf{X}_i) \| p(\mathbf{Z}_i)), \quad [9]$$

where  $\Theta$  represents all unknown parameters, including  $\mathcal{U}$ ,  $\boldsymbol{\pi}$ , the unknown weights in the encoder and the decoder, and the dispersion parameters  $\theta_g$ . The tuning parameter  $\beta$ , introduced in (7) as the  $\beta$ -VAE, is to balance the reconstruction error and regularization. To better adjust for the confounding  $\mathbf{X}_i$  and increase robustness, we add three extra (optional) penalties to obtain our final loss function aggregated over all  $N$  cells as:

$$\begin{aligned} \mathcal{L}(\mathbf{Y}_1, \dots, \mathbf{Y}_N; \mathbf{X}_1, \dots, \mathbf{X}_N, \Theta) &= - (1 - \alpha) \sum_{i=1}^N \mathbb{E}_{q(\mathbf{Z}_i | \mathbf{Y}_i, \mathbf{X}_i)} \log p(\mathbf{Y}_i | \mathbf{Z}_i, \mathbf{X}_i) \\ &\quad - \alpha \sum_{i=1}^N \log p(\mathbf{Y}_i | \mathbf{Z}_i = \mathbf{0}_d, \mathbf{X}_i) + \beta \sum_{i=1}^N D_{\text{KL}}(q(\mathbf{Z}_i | \mathbf{Y}_i, \mathbf{X}_i) \| p(\mathbf{Z}_i)) \\ &\quad + \gamma \Omega_{\text{Jacobian}}(\mathbf{Y}_1, \dots, \mathbf{Y}_N; \mathbf{X}_1, \dots, \mathbf{X}_N) + \kappa \Omega_{\text{MMD}}(\mathbf{Y}_1, \dots, \mathbf{Y}_N; \mathbf{X}_1, \dots, \mathbf{X}_N). \end{aligned} \quad [10]$$

where  $\alpha$  is a tuning parameter for ‘‘soft’’ batch adjustment (8),  $\Omega_{\text{MMD}}$  is the MMD loss to remove the difference of cells from different sources, and  $\Omega_{\text{Jacobian}}$  is the Jacobian regularizer to stabilize the optimization processes (see Section S2.1 for more

146 details). Setting  $\alpha > 0$  ( $\alpha = 0.1$  by default) encourages our decoder to reconstruct  $\mathbf{Y}_i$  using only information from  $\mathbf{X}_i$ , which  
 147 can help to decorrelate  $\mathbf{X}_i$  from  $\mathbf{Z}_i$  so that one can remove the confounding effects of  $\mathbf{X}_i$  in  $\mathbf{Z}_i$  more thoroughly. We set  
 148  $\gamma = 1, \kappa = 0$  as default.

149 All the terms in Eq. (10) can be approximated by the Monte Carlo method efficiently, as we show in Section S2.1. Specifically,  
 150 though the marginal density function  $p(\mathbf{Z}_i)$  is involved in a complex hierarchical mixture model Eq. (6), it still has a closed-form  
 151 representation. The optimization minimizing the loss function Eq. (10) can be efficiently done via stochastic backpropagation  
 152 (9) (10) on mini-batches of data, with the commonly used amortized variation inference (11) for VAE.

153 **S1.2. Trajectory Inference from Posterior Approximations.** After the training step, the model returns the estimated parameters,  
 154 including  $\hat{\mathbf{U}}, \hat{\boldsymbol{\pi}}$ , the encoder  $\hat{q}(\mathbf{Z}_i|\mathbf{Y}_i, \mathbf{X}_i)$ , and the decoder  $\hat{f}(\mathbf{Z}_i|\mathbf{Y}_i, \mathbf{X}_i)$ . Replacing the true posterior density  $p(\mathbf{Z}_i|\mathbf{Y}_i, \mathbf{X}_i)$   
 155 with the approximate posterior density  $\hat{q}(\mathbf{Z}_i|\mathbf{Y}_i, \mathbf{X}_i)$ , we can also obtain an approximation of the posterior distribution of  
 156  $(w_i, c_i)$  for each cell. Specifically, let  $z_i^{(1)}, \dots, z_i^{(L)}$  be  $L$  ( $L = 300$  by default) random samples from  $\hat{q}(\mathbf{Z}_i|\mathbf{Y}_i, \mathbf{X}_i)$ , and use the  
 157 fact that  $(w_i, c_i) \perp\!\!\!\perp (\mathbf{Y}_i, \mathbf{X}_i)|\mathbf{Z}_i$ , we can approximate the posterior density of  $(w_i, c_i)$  as

$$158 \quad \hat{p}(w, c|\mathbf{Y}_i, \mathbf{X}_i) = \frac{1}{L} \sum_{l=1}^L \hat{p}(w, c|\mathbf{Z}_i = z_i^{(l)})$$

159 where  $\hat{p}(w, c|\mathbf{Z}_i)$  is obtained by plugging in  $\hat{\mathbf{U}}$  and  $\hat{\boldsymbol{\pi}}$  into the true posterior density of  $(w_i, c_i)$  given  $\mathbf{Z}_i$ . Similarly, we can get  
 160 the approximate posterior distributions of  $c_i$ , the edge or vertex the cell belongs to, as

$$161 \quad \hat{p}(c_i = c|\mathbf{Y}_i, \mathbf{X}_i) = \frac{1}{L} \sum_{l=1}^L \hat{p}(c_i = c|\mathbf{Z}_i = z_i^{(l)})$$

162 For the cell position  $\tilde{\mathbf{w}}_i$  on the graph  $\mathcal{G}$ , as it is a function of  $w_i$  and  $c_i$  as defined in Eq. (6), we can also efficiently obtain the  
 163 mean  $\boldsymbol{\mu}_{\tilde{\mathbf{w}}_i}$  and the diagonal elements of the covariance matrix  $\Sigma_{\tilde{\mathbf{w}}_i}$  of its posterior distribution  $\hat{p}(\tilde{\mathbf{w}}_i|\mathbf{Y}_i, \mathbf{X}_i)$ . For details in  
 164 calculating the approximate posterior distributions, see Section S2.2. Now, we discuss how to infer the trajectory backbone and  
 165 cell positions along the trajectory with these posterior approximations.

166 **Infer the trajectory backbone  $\mathcal{B}$**  The total number of categories  $K = O(k^2)$  can be large, even with a moderate number of  $k$ .  
 167 While the trajectory backbone  $\mathcal{B}$  typically only sparsely involves a few edges, the estimated  $\hat{\boldsymbol{\pi}}$  may be dense on the unpruned  
 168 entries. Inspired by (12) on Bayesian Gaussian mixture models, to encourage sparsity, we infer the nonzero edges a posteriori  
 169 from the data. Specifically, we define a score for each edge, quantifying the strength of the evidence that the edge exists:

$$170 \quad s_{j_1 j_2} = \frac{|\{i : c_i = C_{j_1 j_2}\}|}{|\{i : \mathbf{e}_{j_1}^\top \tilde{\mathbf{w}}_i > 0.5 \text{ or } \mathbf{e}_{j_2}^\top \tilde{\mathbf{w}}_i > 0.5\}|}$$

171 where the denominator is added to make sure that we can capture the continuous transitions between cell states even when  
 172 these states only involve a small proportion of cells in the cell population. Even though  $\hat{\boldsymbol{\pi}}$  is dense,  $s_{j_1 j_2}$  can be significantly  
 173 nonzero for much fewer edges. From another point of view, an edge score is some ‘‘test statistics’’ to evaluate whether the  
 174 edge exists. To make our algorithm scalable, in practice, instead of obtaining the posterior distributions of  $s_{j_1 j_2}$  to determine  
 175 whether it is significantly nonzero or not, we simply use a deterministic version of the edge score as

$$176 \quad \tilde{s}_{j_1 j_2} = \frac{|\{i : \tilde{c}_i = C_{j_1 j_2}\}|}{|\{i : \mathbf{e}_{j_1}^\top \boldsymbol{\mu}_{\tilde{\mathbf{w}}_i} > 0.5 \text{ or } \mathbf{e}_{j_2}^\top \boldsymbol{\mu}_{\tilde{\mathbf{w}}_i} > 0.5\}|}$$

177 where  $\tilde{c}_i = \arg \max_{c \in \{1, 2, \dots, K\}} \hat{p}(c_i = c|\mathbf{Y}_i, \mathbf{X}_i)$  and  $\boldsymbol{\mu}_{\tilde{\mathbf{w}}_i}$  is the approximate posterior mean of  $\tilde{\mathbf{w}}_i$ . Other edge score choices  
 178 are discussed in Section S2.3.

179 A larger  $\tilde{s}_{j_1 j_2}$  indicates higher confidence assuring that the edge exists in the trajectory backbone  $\mathcal{B}$ . In practice, we include  
 180 an edge  $(j_1, j_2)$  into the estimated backbone  $\hat{\mathcal{B}}$  if  $\tilde{s}_{j_1 j_2} \geq s_0$ , where the cutoff  $s_0$  is 0.01 by default. When we are certain that  
 181 there are no loops in the trajectory, inspired by (2), we further prune  $\hat{\mathcal{B}}$  as the MST of the unpruned graph with  $\tilde{s}_{j_1 j_2}$  as edge  
 182 weights. This typically results in a cleaner shape of our estimated trajectory.

183 **Project cells onto the inferred trajectory backbone** To obtain the position of each cell on the inferred trajectory, we further project  
 184  $\tilde{\mathbf{w}}_i$  onto  $\hat{\mathcal{B}}$ . Given the approximate posterior distributions of  $\tilde{\mathbf{w}}_i$ , we would find a point estimate for the best position of each  
 185 cell  $i$  on  $\hat{\mathcal{B}}$ . Specifically, for each cell, we aim to solve the following optimization problem:

$$186 \quad \hat{\mathbf{w}}_i = \arg \min_{\mathbf{w}} \mathbb{E}_{\hat{p}(\tilde{\mathbf{w}}_i|\mathbf{Y}_i, \mathbf{X}_i)} \|\tilde{\mathbf{w}}_i - \mathbf{w}\|_2^2$$

$$187 \quad \text{s.t.} \quad \text{support}(\mathbf{w}) \subseteq \hat{\mathcal{B}}, \|\mathbf{w}\|_1 = 1, \mathbf{w} \succeq \mathbf{0}. \quad [11]$$

188 Since the support of  $\tilde{\mathbf{w}}_i$  is restricted to the inferred trajectory backbone  $\hat{\mathcal{B}}$ , only one or two entries of  $\hat{\mathbf{w}}_i$  can be nonzero,  
 189 depending on whether the cell is at the vertex or edge of  $\hat{\mathcal{B}}$ . Though the  $L_2$  loss in the objective function of optimization  
 190 problem Eq. (11) is not the only choice, it can result in a closed-form solution that allows fast computation.

191 **Proposition 1.** *The optimization problem (11) is equivalent to finding*

$$192 \quad \hat{\mathbf{w}}_i = \arg \min_{\mathbf{w}} \quad \|\boldsymbol{\mu}_{\tilde{\mathbf{w}}_i} - \mathbf{w}\|_2^2$$

$$193 \quad \text{s.t.} \quad \text{support}(\mathbf{w}) \subseteq \hat{\mathcal{B}}, \|\mathbf{w}\|_1 = 1, \mathbf{w} \succeq \mathbf{0}, \quad [12]$$

194 where  $\boldsymbol{\mu}_{\tilde{\mathbf{w}}_i}$  is the mean of  $\hat{p}(\tilde{\mathbf{w}}_i | \mathbf{Y}_i, \mathbf{X}_i)$ . Denote the  $j$ th component of  $\boldsymbol{\mu}_{\tilde{\mathbf{w}}_i}$  as  $\mu_j$  and let  $\mathcal{EN}(\hat{\mathcal{B}}) = \mathcal{E}(\hat{\mathcal{B}}) \cup \{(j, j) : j \in \mathcal{N}(\hat{\mathcal{B}})\}$ ,  
 195 then the best projection is given by

$$196 \quad (j_1^*, j_2^*) \in \arg \max_{(j_1, j_2) \in \mathcal{EN}(\hat{\mathcal{B}})} (\mu_{j_1} - \mu_{j_2})^2 + 2(\mu_{j_1} + \mu_{j_2}) - \mathbb{1}_{\{j_1=j_2\}},$$

197 and the corresponding solution  $\text{proj}_{(j_1^*, j_2^*)}(\boldsymbol{\mu}_{\tilde{\mathbf{w}}_i})$  has entries

$$198 \quad [\text{proj}_{(j_1^*, j_2^*)}(\boldsymbol{\mu}_{\tilde{\mathbf{w}}_i})]_j = \begin{cases} \mu_{j_1^*} + \frac{1}{2}(1 - \mu_{j_1^*} - \mu_{j_2^*} + \mathbb{1}_{\{j_1^*=j_2^*\}}) & j \in \{j_1^*, j_2^*\} \\ 0 & \text{otherwise,} \end{cases}$$

199 for  $j = 1, \dots, k$ . The best projection reduces to be a vertex if  $j_1^* = j_2^*$ .

200 The proof of Proposition 1 is included in Section S2.4. Intuitively, finding the optimal projection of  $\tilde{\mathbf{w}}_i$  minimizing the  $L_2$   
 201 loss is equivalent to simply projecting the posterior mean  $\boldsymbol{\mu}_{\tilde{\mathbf{w}}_i}$ . In addition, as shown in our proof of Proposition 1, most cells  
 202 will either project onto an edge or an isolated vertex. Notice that the solution for the optimization problem Eq. (12) may not  
 203 be unique. However, it is generally unique due to floating-point computation in practice.

204 Next, we also want to quantify the uncertainty of the projected position  $\hat{\mathbf{w}}_i$  as an estimate of  $\tilde{\mathbf{w}}_i$ . A general metric to evaluate  
 205 the uncertainty is given by  $\mathbb{E}_{\hat{p}(\tilde{\mathbf{w}}_i | \mathbf{Y}_i, \mathbf{X}_i)}[d(\hat{\mathbf{w}}_i, \tilde{\mathbf{w}}_i)]$ , where  $d(\cdot, \cdot)$  is a metric or distance function. When  $d(\mathbf{w}, \mathbf{w}') = \|\mathbf{w} - \mathbf{w}'\|_2^2$   
 206 is also the  $L_2$  loss, we obtain the projection mean square error (MSE) as

$$207 \quad \mathbb{E}_{\hat{p}(\tilde{\mathbf{w}}_i | \mathbf{Y}_i, \mathbf{X}_i)}[d(\hat{\mathbf{w}}_i, \tilde{\mathbf{w}}_i)] = \|\hat{\mathbf{w}}_i - \boldsymbol{\mu}_{\tilde{\mathbf{w}}_i}\|_2^2 + \text{tr}(\Sigma_{\tilde{\mathbf{w}}_i}), \quad [13]$$

208 which is easily computable. Notice that our projection MSE ignores the uncertainty in  $\hat{\mathcal{B}}$  and the approximation error in  
 209  $\hat{p}(\tilde{\mathbf{w}}_i | \mathbf{Y}_i, \mathbf{X}_i)$ , so it is an underestimate of the true uncertainty. However, we think that the relative magnitude of our projection  
 210 MSE would still be a useful quantity to compare the projection accuracy across cells. Another pattern is that the projection  
 211 MSE is typically smaller when the cells are near vertices. We provide our understanding and a detailed discussion of this  
 212 pattern in Section S1.5.

213 Finally, we obtain the point estimate of each cell's pseudotime  $T_i$  defined in Eq. (2). Given the inferred trajectory  $\hat{\mathcal{B}}$ ,  
 214 the user can assign a root vertex based on prior biological knowledge. We also provide an automatic root selection step  
 215 following Tempora (13) when cells are collected from a series of time points. The idea is to choose the vertex with the earliest  
 216 collection time as the root. Specifically, let  $r_i$  be the collection time of cell  $i$ , we calculate the ‘‘collection time’’ of vertex  $j$  as  
 217  $\omega_j = \sum_i \hat{\mathbf{w}}_{ij} r_i / \sum_i \hat{\mathbf{w}}_{ij}$ , a weighted average of the collection time of the cells near the vertex. A vertex  $k_0 \in \mathcal{N}(\hat{\mathcal{B}})$  is chosen as  
 218 the root if it has the smallest collection time  $\omega_j$  and is not an isolated vertex.

219 Once the root vertex of the trajectory backbone  $\hat{\mathcal{B}}$  is obtained, it is straightforward to obtain  $\hat{\mathbf{o}}$ , the estimated pseudotime  
 220 of the vertices, by plugging in  $\hat{\mathcal{B}}$  into the definition of  $\mathbf{o}$  in Eq. (1). Then, a point estimate of the pseudotime  $T_i$  of cell  $i$  can be  
 221 given as  $\hat{T}_i = \hat{\mathbf{o}}^\top \hat{\mathbf{w}}_i$ .

222 **S1.3. Differential gene expression along the trajectory.** We provide a polynomial regression approach to find differentially  
 223 expressed genes along our inferred trajectory backbone. We focus on finding genes that are associated with the pseudotime  
 224 ordering after adjusting for confounding covariates and provide a scalable way to obtain the  $p$ -values of the genes, taking into  
 225 consideration that the pseudotimes of the cells are estimated.

226 In particular, to find genes that are differentially expressed along the pseudotime ordering for a subset of cells  $\mathcal{S}$ , we work  
 227 with the following polynomial regression for each gene  $g$ :

$$228 \quad Y_{ig} = \beta_{0g} + \beta_{1g} \text{rank}(T_i) + \dots + \beta_{Kg} \text{rank}(T_i)^K + \mathbf{X}_i^\top \boldsymbol{\beta}_{Xg} + e_{ig}, \quad \mathbb{E}(e_{ig}) = 0, \forall i \in \mathcal{S} \quad [14]$$

229 where  $Y_{ig}$  is the log-transformed and normalized count for cell  $i$  and gene  $g$ ,  $\text{rank}(T_i)$  is the rank of true pseudotime  $T_i$  of cell  
 230  $i$ , and the linear term  $\mathbf{X}_i^\top \boldsymbol{\beta}_{Xg}$  is to adjust for the confounding effects of known covariates. We allow  $e_{ig}$ 's to have unequal  
 231 variances as  $Y_{ig}$ 's are log-transformed and normalized counts.  $K$  is the degree of the polynomials and by default, is chosen as 2  
 232 to allow both linear and quadratic change of the gene expression along pseudotimes. For each gene, we aim to test for the  
 233 global null  $H_{0g} : \beta_{1g} = \dots = \beta_{Kg} = 0$ . We use  $\text{rank}(T_i)$  in the regression model so that the results will not be affected by the  
 234 scaling of pseudotimes.

235 The challenge here is that the true pseudotime  $T_i$  is not observed. Instead, we only have estimated pseudotime  $\hat{T}_i$  from the  
 236 data, which has unknown uncertainty and is correlated with  $Y_{ig}$ . As a consequence, the  $p$ -value for  $H_{0g}$  ignoring the fact that  
 237 the pseudotimes of the cells are estimated is likely invalid. To adjust for this, we take a simple  $p$ -value calibration approach  
 238 following (14). To get an initial  $p$ -value, we plug in  $\hat{T}_i$  after normalizing and centering each  $\text{rank}(T_i)^k$ , and we estimate the

239 coefficients  $\beta_{kg}$  via the ordinary least squares. Treating  $\widehat{T}_i$  as the true  $T_i$ , we obtain the variances of  $\widehat{\beta}_{kg}$  through the sandwich  
 240 estimator, allowing for heterogeneity in  $e_{ig}$ . The  $t$ -statistics is then defined as  $t_{kg} = \widehat{\beta}_{kg} / \widehat{\text{sd}}(\widehat{\beta}_{kg})$ . To calibrate the  $p$ -values,  
 241 instead of assuming  $t_{kg} \sim \mathcal{N}(0, 1)$  under the null, we assume that  $t_{kg} \sim \mathcal{N}(0, \sigma_k^2)$  where  $\sigma_k^2$  is estimated from the median  
 242 absolute deviation (MAD) of  $\{t_{kg}, g = 1, 2, \dots, G\}$ . Then we use the Bonferroni combination approach across  $k$  to obtain  
 243 a combined  $p$ -value for the null  $H_{0g}$ . To select differentially expressed genes, these  $p$ -values are further adjusted with the  
 244 Benjamini-Hochberg procedure for multiple testing corrections.

245 There are other approaches to detecting differentially expressed genes along estimated pseudotime. For instance, tradeSeq  
 246 (15) employs a generalized linear regression model on raw scRNA-seq counts, assuming a Negative Binomial distribution of  
 247 the data. They regress  $Y_{ig}$  on the smoothing splines of  $\widehat{T}_i$ . PseudotimeDE (16) uses a sophisticated approach to calibrate  
 248 the  $p$ -values in differential testing with estimated pseudotime. To adjust for the bias treating  $\widehat{T}_i$  as the true pseudotime,  
 249 PseudotimeDE employs cell subsampling and permutation of estimated pseudotime to generate a null distribution for the  
 250 regression test statistics of each gene. Compared to tradeSeq and pseudotimeDE, our approach sacrifices some rigor for  
 251 increased computational efficiency. See Section S3.3 for an empirical comparison of our differential testing approach with  
 252 PseudotimeDE and tradeSeq.

253 **S1.4. Model estimation with practical considerations.** Instead of using all the genes, by default, we select highly variable genes  
 254 and preprocess the gene expressions by scanpy (17). Then the normalized, log-transformed, and scaled gene expressions are  
 255 provided as the inputs of our VAE.

256 As the optimization of our loss Eq. (10) generally results in a locally optimal solution, we need a good initialization of our  
 257 parameters, especially for  $\mathbf{U}$  and  $\boldsymbol{\pi}$  defined in Eq. (3) and Eq. (6). Also, our framework requires a pre-determined number of  
 258 states  $k$ . Inspired by other existing TI methods, we design a three-step algorithm for model initialization and estimation.

259 The first step is pretraining, where we train the model with  $\beta = 0$ , to only minimize the reconstruction loss, which does not  
 260 involve the unknown parameters  $\boldsymbol{\pi}$  and  $\mathbf{U}$ . The Jacobian and MMD regularizers can also be used in this stage. The purpose of  
 261 this step is to obtain better weights for the encoder and the decoder and to get an initial low-dimensional representation, which  
 262 can be used to initialize  $\mathbf{U}$  and determine  $k$ .

263 The second step is to initialize the latent space after pretraining. In our experiments, we use annotated cell types to initialize  
 264 the latent space and set  $k$  to be the number of cell types. However, one can also perform cell clustering with the Louvain  
 265 algorithm (18) on the estimated posterior means  $\widehat{\boldsymbol{\mu}}_{\mathbf{Y}_i, \mathbf{X}_i}$  of  $\mathbf{Z}_i$  to determine  $k$ . We initialize  $\mathbf{U}$  with the cluster centers. As  $\boldsymbol{\pi}$   
 266 involves both the  $k$  vertices and  $k(k-1)/2$  edges, we have no information yet and just uniformly initialize  $\boldsymbol{\pi}$  in this step. We  
 267 introduce a pruning mechanism for better estimating  $\boldsymbol{\pi}$ . Specifically, we set those  $\pi_i$ 's corresponding to edges with the top 50%  
 268 lengths (the distance between the initial centers) as 0 and freeze them when training.

269 The last step is to train our whole network, optimizing the loss function Eq. (10) with  $\beta = 1$ . During both the pretraining  
 270 and training steps, the optimization is early stopped when the evaluation loss decreases sufficiently slowly.

271 Finally, to make VITAE scalable to handle large datasets and have a comparable computational cost as other TI methods,  
 272 we accelerate VITAE for large datasets by reducing the dimension of the input. We replace the  $G$  features with its top  $R$   
 273 ( $R = 64$  by default) principle component (PC) scores  $\mathbf{F}_i$ . The output of the decoder is also replaced by the reconstruction of  $\mathbf{F}_i$ ,  
 274 and the likelihood of  $p(\mathbf{Y}_i | \mathbf{Z}_i, \mathbf{X}_i)$  in the loss function Eq. (10) is replaced by Gaussian densities assumed on  $\mathbf{F}_i$  (Section S2.1).  
 275 As we will show in the Result Section, the accelerated VITAE with Gaussian densities provides results comparable to our  
 276 original likelihood-based VITAE in general, while it can be much faster than the original scheme. By default, both the encoder  
 277 and decoder have one hidden layer, with 32 and 16 units for the accelerated and likelihood-based VITAE, respectively. The  
 278 bottleneck layer only has a dimension of 8. The hidden layers are all fully connected with the leaky rectified linear activation  
 279 function (19) and Batch Normalization (20).

280 **S1.5. Discussion on Uncertainty Quantification.** We observed in practice that our calculated projection MSE is always smaller  
 281 for the cells near vertices. Why do we observe such a pattern? Is it due to the estimation bias in our approximation of  
 282 the posterior distribution, or is it an intrinsic property of the L2 loss under our mixture model? In order to answer these  
 283 questions, we consider the case where we observe the latent space  $\mathbf{Z}_i$ , so that we can compute the true posterior distributions  
 284 of  $\mathbf{Z}_i | \tilde{\boldsymbol{w}}_i$ . Also, we focus on  $\text{tr}(\Sigma_{\tilde{\boldsymbol{w}}_i})$ , as from our observations it is typically the leading term in (12). Specifically, we define

285 our hierarchical model with observed  $\mathbf{Z}_i$  following (3) and (5) by setting  $d = 2$ ,  $k = 3$  with  $\mathbf{U} = \begin{pmatrix} 0 & 0.5 & 2 \\ 2 & 0.5 & 0 \end{pmatrix}$ . The defined  
 286 backbone is shown in Figure S0.1a and we choose vertex 1 as the root of the trajectory.

287 First, we take  $\boldsymbol{\pi} = (\frac{1}{5} \ \frac{1}{5} \ 0 \ \frac{1}{5} \ \frac{1}{5} \ \frac{1}{5})$ , so that the edges and vertices have equal probabilities. As  $\mathbf{Z}_i$  is observed, we  
 288 can compute the true posterior mean and variances from Section S2.2 for any given  $\mathbf{Z}_i$ . Here we show the posterior variances  
 289 given  $\mathbf{Z}_i = \mathbf{U}\tilde{\boldsymbol{w}}_i$  which are exactly on the trajectory, and discuss how the variances change with  $T_i = \boldsymbol{o}^\top \tilde{\boldsymbol{w}}_i$ . The cell  $i$  is at the  
 290 vertices when  $T_i = 0, 1$  or  $2$ . Figure S0.1b shows how the posterior variances of each  $\tilde{w}_{ij}$  change with  $T_i$ . The leading term of  
 291 the projection MSE  $\text{tr}(\Sigma_{\tilde{\boldsymbol{w}}_i})$  summing up the three terms in Figure S0.1b has an M-shape as shown in Figure S0.1c, indicating  
 292 that the cells indeed have smaller projection uncertainties when they are near the vertices.

293 One explanation of the above phenomenon is that as there are nonzero probabilities exactly at the vertices, the distribution  
 294 is denser near the vertices. As a consequence, there is less uncertainty on  $\tilde{\boldsymbol{w}}_i$  if it is closer to a vertex. However, what we find  
 295 surprising is that when we set  $\boldsymbol{\pi} = (0 \ \frac{1}{2} \ 0 \ 0 \ \frac{1}{2} \ 0)$ , where the cells can only be on the edges, we still observe an M-shape  
 296 for the change of  $\text{tr}(\Sigma_{\tilde{\boldsymbol{w}}_i})$  along  $T_i$ . In other words, the projection MSE from true posteriors is smaller near the vertices no

297 matter whether the vertices have nonzero probabilities or not, though the difference is smaller when the vertices have zero probabilities. As a consequence, we believe that this pattern is an intrinsic property of the  $L_2$  loss under our mixture model.

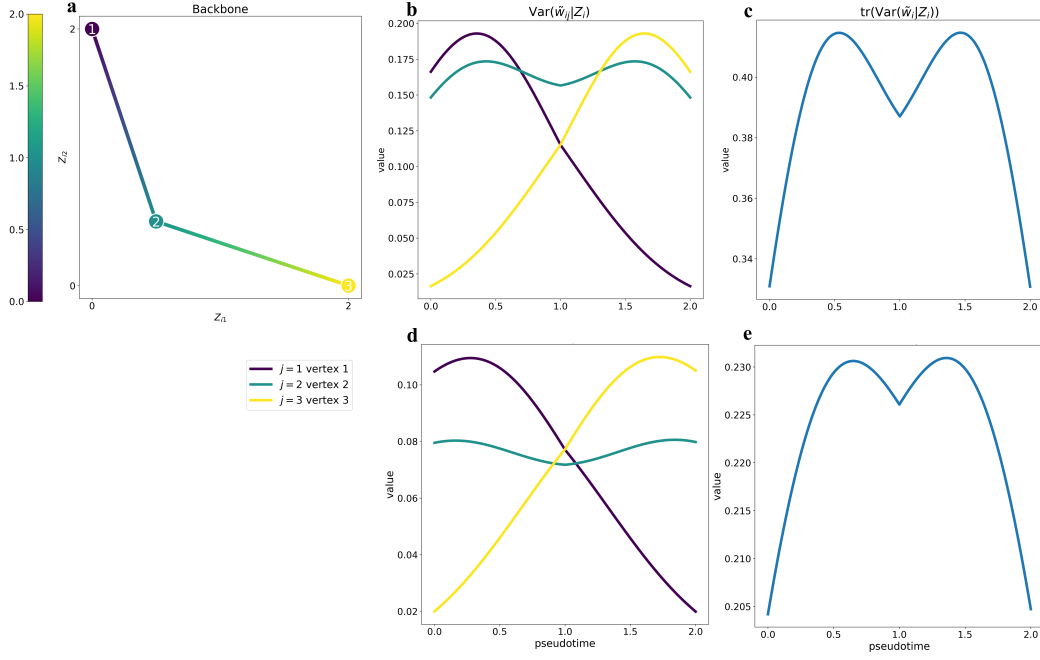


Fig S0.1. (a) The underlying backbone of the assumed model colored by pseudotime. (b) The values of  $\text{Var}(\hat{w}_{ij}|\mathbf{Z}_i)$  along the trajectory and (c) the values of  $\text{tr}(\text{Var}(\hat{w}_{ij}|\mathbf{Z}_i))$  along the trajectory are for the case when  $\pi = (0.2, 0.2, 0, 0.2, 0.2, 0.2)$ . (d-e) are the same as (b-c) for the case when  $\pi = (0, 0.5, 0, 0, 0.5, 0)$ .

298

## 299 S2. Technical details

### 300 S2.1. Loss Function.

301 **Basic loss function.** Note that the loss for each cell in Eq. (10) takes the form

$$\begin{aligned}
 \mathcal{L}(\mathbf{Y}_i; \mathbf{X}_i, \Theta) &= -(1 - \alpha) \mathbb{E}_{q(\mathbf{Z}_i|\mathbf{Y}_i, \mathbf{X}_i)} \log p(\mathbf{Y}_i|\mathbf{Z}_i, \mathbf{X}_i) \\
 &\quad - \alpha \log p(\mathbf{Y}_i|\mathbf{Z}_i = \mathbf{0}_d, \mathbf{X}_i) + \beta D_{\text{KL}}(q(\mathbf{Z}_i|\mathbf{Y}_i, \mathbf{X}_i) \| p(\mathbf{Z}_i)) \\
 &= -(1 - \alpha) \mathbb{E}_{q(\mathbf{Z}_i|\mathbf{Y}_i, \mathbf{X}_i)} \log p(\mathbf{Y}_i|\mathbf{Z}_i, \mathbf{X}_i) - \alpha \log p(\mathbf{Y}_i|\mathbf{Z}_i = \mathbf{0}_d, \mathbf{X}_i) \\
 &\quad + \beta \mathbb{E}_{q(\mathbf{Z}_i|\mathbf{Y}_i, \mathbf{X}_i)} \log q(\mathbf{Z}_i|\mathbf{Y}_i, \mathbf{X}_i) - \beta \mathbb{E}_{q(\mathbf{Z}_i|\mathbf{Y}_i, \mathbf{X}_i)} \log p(\mathbf{Z}_i),
 \end{aligned}$$

306 when  $\Omega_{\text{MMD}} \equiv \Omega_{\text{Jacobian}} \equiv 0$ . Next, we show how to compute the above terms efficiently.

307 1.  $\mathbb{E}_{q(\mathbf{Z}_i|\mathbf{Y}_i, \mathbf{X}_i)} \log p(\mathbf{Y}_i|\mathbf{Z}_i, \mathbf{X}_i)$

$$\mathbb{E}_{q(\mathbf{Z}_i|\mathbf{Y}_i, \mathbf{X}_i)} \log p(\mathbf{Y}_i|\mathbf{Z}_i, \mathbf{X}_i) \approx \frac{1}{L} \sum_{l=1}^L \sum_{g=1}^G \log p(Y_{ig}|\mathbf{Z}_i = \mathbf{z}_i^{(l)}, \mathbf{X}_i)$$

309 where  $\mathbf{z}_i^{(1)}, \dots, \mathbf{z}_i^{(L)}$  are Monte Carlo samples from the approximate posterior distribution  $q(\mathbf{Z}_i|\mathbf{Y}_i, \mathbf{X}_i)$ . Here  $p(Y_{ig}|\mathbf{Z}_i, \mathbf{X}_i)$  depends on the distribution assumptions of  $Y_{ig}$ .

311 (1) For scRNA-seq with UMI,  $Y_{ig}|\mathbf{Z}_i, \mathbf{X}_i$  follows a Negative Binomial distribution  $\mathcal{NB}(\lambda_{ig}, \theta_g)$  with a probability mass function,

$$p(y|\mathbf{Z}_i, \mathbf{X}_i) = \binom{\theta_g + y - 1}{y} \left( \frac{\theta_g}{\lambda_{ig} + \theta_g} \right)^{\theta_g} \left( \frac{\lambda_{ig}}{\lambda_{ig} + \theta_g} \right)^y,$$

314 for  $i = 1, \dots, N$ . It is noted that only  $\lambda_{ig}$  dependent on  $\mathbf{Z}_i$  and  $\mathbf{X}_i$ .

(2) For non-UMI data,  $Y_{ig}|\mathbf{Z}_i, \mathbf{X}_i$  follows a zero-inflated Negative Binomial distribution  $\mathcal{ZINB}(\lambda_{ig}, \theta_g, \phi_{ig})$  with the probability mass function,

$$p(y|\mathbf{Z}_i, \mathbf{X}_i) = \begin{cases} \phi_{ig} + (1 - \phi_{ig}) p_{NB}(0; \lambda_{ig}, \theta_g) & \text{if } y = 0 \\ (1 - \phi_{ig}) p_{NB}(y; \lambda_{ig}, \theta_g) & \text{if } y > 0, \end{cases}$$

where  $p_{NB}(y; \lambda_{ig}, \theta_g)$  is the probability mass function of  $\mathcal{NB}(\lambda_{ig}, \theta_g)$ .

(3) For the accelerated VITAE with a Gaussian model,  $\mathbf{Y}_i$  is replaced by the PC scores  $\mathbf{F}_i$ . We assume that each  $F_{ig}|\mathbf{Z}_i, \mathbf{X}_i$  follows a Gaussian distribution  $\mathcal{N}(\nu_{ig}, \tau_i^2)$  with probability density function,

$$p(y|\mathbf{Z}_i, \mathbf{X}_i) = \frac{1}{\sqrt{2\pi\tau_i}} \exp\left(-\frac{(y - \nu_{ig})^2}{2\tau_i^2}\right).$$

2.  $\mathbb{E}_{q(\mathbf{Z}_i|\mathbf{Y}_i, \mathbf{X}_i)} \log p(\mathbf{Z}_i)$

$$\mathbb{E}_{q(\mathbf{Z}_i|\mathbf{Y}_i, \mathbf{X}_i)} \log p(\mathbf{Z}_i) = \mathbb{E}_{q(\mathbf{Z}_i|\mathbf{Y}_i, \mathbf{X}_i)} \left[ \log \left( \sum_{c=1}^K p(\mathbf{z}|c_i = c)p(c) \right) \right] \approx \frac{1}{L} \sum_{l=1}^L \log \left( \sum_{c=1}^K p(\mathbf{z}^{(l)}|c_i = c)p(c) \right).$$

When  $c \in \mathcal{N}(\mathcal{G})$ , the conditional density of  $\mathbf{Z}_i|c_i = c$  is given by

$$p(\mathbf{z}|c_i = c) = \int_0^1 p(\mathbf{z}|c_i = c, w)dw = \frac{1}{(2\pi)^{\frac{d}{2}}} \int_0^1 e^{-\frac{1}{2}(\mathbf{z} - \mathbf{U}b_c)^\top (\mathbf{z} - \mathbf{U}b_c)} dw = \varphi_d(\mathbf{U}b_c).$$

where  $\varphi_d(\cdot)$  is the probability density function of the standard  $d$ -dimensional multivariate Gaussian distribution.

When  $c \in \mathcal{E}(\mathcal{G})$ , the conditional density of  $\mathbf{Z}_i|c_i = c$  is given by

$$\begin{aligned} p(\mathbf{z}|c_i = c) &= \int_0^1 p(\mathbf{z}|c_i = c, w)dw \\ &= \frac{1}{(2\pi)^{\frac{d}{2}}} \int_0^1 e^{-\frac{1}{2}(\mathbf{z} - \mathbf{U}\bar{w})^\top (\mathbf{z} - \mathbf{U}\bar{w})} d\bar{w} \\ &= \frac{\frac{\boldsymbol{\alpha}_{zc} = \mathbf{U}(b_c - a_c)}{\boldsymbol{\beta}_{zc} = \mathbf{z} - \mathbf{U}b_c}}{(2\pi)^{\frac{d}{2}}} \int_0^1 e^{-\frac{1}{2}(w\boldsymbol{\alpha}_{zc} + \boldsymbol{\beta}_{zc})^\top (w\boldsymbol{\alpha}_{zc} + \boldsymbol{\beta}_{zc})} dw \\ &= \frac{1}{(2\pi)^{\frac{d}{2}}} \int_0^1 e^{-\frac{1}{2}(\boldsymbol{\alpha}_{zc}^\top \boldsymbol{\alpha}_{zc} w^2 + 2\boldsymbol{\alpha}_{zc}^\top \boldsymbol{\beta}_{zc} w + \boldsymbol{\beta}_{zc}^\top \boldsymbol{\beta}_{zc})} dw \\ &= \frac{\sigma_c e^{\frac{t_{zc}}{2}}}{(2\pi)^{\frac{d-1}{2}}} \left[ \Phi\left(\frac{1 - \nu_{zc}}{\sigma_c}\right) - \Phi\left(-\frac{\nu_{zc}}{\sigma_c}\right) \right], \end{aligned}$$

with

$$\nu_{zc} = -\frac{\boldsymbol{\alpha}_{zc}^\top \boldsymbol{\beta}_{zc}}{\boldsymbol{\alpha}_{zc}^\top \boldsymbol{\alpha}_{zc}}, \quad \sigma_{zc}^2 = \frac{1}{\boldsymbol{\alpha}_{zc}^\top \boldsymbol{\alpha}_{zc}}, \quad t_{zc} = -\boldsymbol{\beta}_{zc}^\top \boldsymbol{\beta}_{zc} + \frac{(\boldsymbol{\alpha}_{zc}^\top \boldsymbol{\beta}_{zc})^2}{\boldsymbol{\alpha}_{zc}^\top \boldsymbol{\alpha}_{zc}}$$

where  $\Phi(\cdot)$  is the cumulative density function of the standard Gaussian distribution.

Therefore, the marginal density of  $\mathbf{Z}_i$  is given by

$$p(\mathbf{z}) = \sum_{c=1}^K p(\mathbf{z}|c_i = c)\pi_c = \sum_{c \in \mathcal{N}(\mathcal{G})} \pi_c \varphi_d(\mathbf{U}b_c) + \sum_{c \in \mathcal{E}(\mathcal{G})} \frac{\pi_c \sigma_{zc} e^{\frac{t_{zc}}{2}}}{(2\pi)^{\frac{d-1}{2}}} \left[ \Phi\left(\frac{1 - \nu_{zc}}{\sigma_{zc}}\right) - \Phi\left(-\frac{\nu_{zc}}{\sigma_{zc}}\right) \right].$$

3.  $\mathbb{E}_{q(\mathbf{Z}_i|\mathbf{Y}_i, \mathbf{X}_i)} \log q(\mathbf{Z}_i|\mathbf{Y}_i, \mathbf{X}_i)$

Since the approximate posterior density  $q(\mathbf{Z}_i|\mathbf{Y}_i, \mathbf{X}_i)$  is  $\mathcal{N}(\boldsymbol{\mu}_{\mathbf{Y}_i, \mathbf{X}_i}, \boldsymbol{\Sigma}_{\mathbf{Y}_i, \mathbf{X}_i})$ , we have

$$\begin{aligned} &\mathbb{E}_{q(\mathbf{Z}_i|\mathbf{Y}_i, \mathbf{X}_i)} \log q(\mathbf{Z}_i|\mathbf{Y}_i, \mathbf{X}_i) \\ &= \mathbb{E}_{q(\mathbf{Z}_i|\mathbf{Y}_i, \mathbf{X}_i)} \left[ -\frac{d}{2} \log(2\pi) - \frac{1}{2} \sum_{i=1}^d \log \sigma_{\mathbf{Y}_i, \mathbf{X}_i, i}^2 - \frac{1}{2} \sum_{j=1}^d \frac{(Z_{ij} - \mu_{\mathbf{Y}_i, \mathbf{X}_i, j})^2}{\sigma_{\mathbf{Y}_i, \mathbf{X}_i, j}^2} \right] \\ &= -\frac{d}{2} \log(2\pi) - \frac{1}{2} \sum_{i=1}^d \log \sigma_{\mathbf{Y}_i, \mathbf{X}_i, i}^2 - \frac{1}{2} \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \left( \sum_{j=1}^d \frac{(Z_{ij} - \mu_{\mathbf{Y}_i, \mathbf{X}_i, j})^2}{\sigma_{\mathbf{Y}_i, \mathbf{X}_i, j}^2} \right) \\ &= -\frac{d}{2} \log(2\pi) - \frac{1}{2} \sum_{i=1}^d (\log \sigma_{\mathbf{Y}_i, \mathbf{X}_i, i}^2 + 1). \end{aligned}$$

344 **Jacobian regularization for stabilizing optimization.** To stabilize the training process and make the model robust against noises, we  
 345 adopt the Jacobian regularization method from (21, 22). Suppose  $\mathbf{Z}_i \in \mathbb{R}^d$ ,  $\mathbf{Y}_i \in \mathbb{R}^G$ ,  $\mathbf{Z}_{ij}$  is the  $j$ -th coordinate of vector  $\mathbf{Z}_i$   
 346 and  $\mathbf{Y}_{ig}$  is the  $g$ -th coordinate of vector  $\mathbf{Y}_i$ . Then we control the Frobenius norm of the Jacobians of the output of the encoder  
 347  $\mathbf{Z}_i$  with respect to the input  $\mathbf{Y}_i$ :

$$\Omega_{\text{Jacobian}}(\mathbf{Y}_1, \dots, \mathbf{Y}_N; \mathbf{X}_1, \dots, \mathbf{X}_N) = \sum_{i=1}^N \sum_{j=1}^d \sum_{g=1}^G \mathbb{E}_{q(\mathbf{Z}_i | \mathbf{Y}_i, \mathbf{X}_i)} \left[ \left( \frac{\partial \mathbf{Z}_{ij}}{\partial \mathbf{Y}_{ig}} \right)^2 \right]. \quad [15]$$

349 **MMD regularization for removing conditional differences.** We can also include the maximum mean discrepancy (MMD) loss as  
 350 utilized in SAUCIE (23) and scArches (24), which explicitly penalizes the distances between cells from any pairings we want to  
 351 adjust in the latent space. Let  $Z_1, Z'_1, Z_2, Z'_2$  be independent samples from two distributions  $\mathcal{P}_1, \mathcal{P}_2$ , then the MMD for  
 352 these two distributions is:

$$\mathcal{D}(\mathcal{P}_1, \mathcal{P}_2) = \mathbb{E}_{Z_1, Z'_1} [k(Z_1, Z'_1)] + \mathbb{E}_{Z_2, Z'_2} [k(Z_2, Z'_2)] - 2\mathbb{E}_{Z_1, Z_2} [k(Z_1, Z_2)]$$

354 where  $k(\cdot, \cdot)$  is a kernel function. Empirically, if we want to integrate data from any two cell groups  $\mathcal{P}_1$  and  $\mathcal{P}_2$  which have  $n_1$   
 355 and  $n_2$  cells respectively, we punish large MMD in the latent space. The MMD loss for two groups can be defined as follows:

$$L_{\text{MMD}}(\mathcal{P}_1, \mathcal{P}_2) = -\frac{1}{n_1^2} \sum_{i_1, i_2 \in \mathcal{P}_1} k(\mathbf{Z}_{i_1}, \mathbf{Z}_{i_2}) - \frac{1}{n_2^2} \sum_{i_1, i_2 \in \mathcal{P}_2} k(\mathbf{Z}_{i_1}, \mathbf{Z}_{i_2}) + \frac{2}{n_1 n_2} \sum_{i_1 \in \mathcal{P}_1, i_2 \in \mathcal{P}_2} k(\mathbf{Z}_{i_1}, \mathbf{Z}_{i_2})$$

where  $\mathbf{Z}_i$  is the latent vector for cell  $i$ . In practice, we employ a multi-scale RBF kernel, proposed in trVAE (25). The multi-scale  
 RBF kernel is defined as  $k(\mathbf{Z}_{i_1}, \mathbf{Z}_{i_2}) = \sum_{\lambda=1}^l k(\mathbf{Z}_{i_1}, \mathbf{Z}_{i_2}, \lambda)$  where  $k(\mathbf{Z}_{i_1}, \mathbf{Z}_{i_2}, \lambda) = \exp(-\lambda \|\mathbf{Z}_{i_1} - \mathbf{Z}_{i_2}\|^2)$  and  $\lambda$  is a hyper-  
 parameter. We set  $l = 19$  by default and let  $\{\lambda_i\}_{i=1}^l = [10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 0.1, 1, 5, 10, 15, 20, 25, 30, 35, 100, 10^3, 10^4, 10^5, 10^6]$ .  
 Then, the MMD penalty for the whole dataset can be defined as:

$$\Omega_{\text{MMD}} = \sum_{(s,t) \in \text{all pairs}} L_{\text{MMD}}(\mathcal{P}_s, \mathcal{P}_t).$$

357 Suppose we wish to apply the MMD penalty to a set of  $k$  cell groups (such as replicates) denoted by  $\{\mathcal{P}_1, \mathcal{P}_2, \dots, \mathcal{P}_k\}$ . We can  
 358 define a set of pairs as  $\text{pairs} = \{(i, j) \mid 1 \leq i < j \leq k\}$  to represent all possible pairwise combinations of the  $k$  groups.

## 359 S2.2. Posterior Estimation.

### 360 1. $\mathbf{Z}_i | \mathbf{Y}_i, \mathbf{X}_i$

361 Since the approximate posterior distribution of  $\mathbf{Z}_i | \mathbf{Y}_i, \mathbf{X}_i$  is  $\mathcal{N}_d(\boldsymbol{\mu}_{\mathbf{Y}_i, \mathbf{X}_i}, \boldsymbol{\Sigma}_{\mathbf{Y}_i, \mathbf{X}_i})$ , we can use  $\boldsymbol{\mu}_{\mathbf{Y}_i, \mathbf{X}_i}$  as the latent  
 362 representation of  $\mathbf{Y}_i$ .

### 363 2. $c_i | \mathbf{Y}_i, \mathbf{X}_i$

364 It is worth noting that a useful property of our model is  $\mathbf{Y}_i \perp\!\!\!\perp (w_i, c_i) | \mathbf{Z}_i, \mathbf{X}_i$ , which is due to the fact that the probability  
 365 mass function of  $\mathbf{Y}_{ig} | \mathbf{Z}_i, w_i, c_i, \mathbf{X}_i$  is  $p(y | \mathbf{Z}_i, w_i, c_i, \mathbf{X}_i) = p(y | \mathbf{Z}_i, \mathbf{X}_i)$ , which only depends on  $\mathbf{Z}_i$  and  $\mathbf{X}_i$ . Note that we  
 366 have already derived the formula for  $p(\mathbf{z} | c_i = c)$  in Section S2.1, we have

$$\begin{aligned} p(c | \mathbf{Y}_i, \mathbf{X}_i) &= \int_{\mathbb{R}} p(c, \mathbf{z} | \mathbf{Y}_i, \mathbf{X}_i) d\mathbf{z} \\ &= \frac{\mathbf{Y}_i | \mathbf{Z}_i, \mathbf{X}_i \perp\!\!\!\perp (w_i, c_i) | \mathbf{Z}_i, \mathbf{X}_i}{\mathbf{Z}_i, c_i \perp\!\!\!\perp \mathbf{X}_i} \int_{\mathbf{z}} p(c | \mathbf{Z}_i = \mathbf{z}) p(\mathbf{z} | \mathbf{Y}_i, \mathbf{X}_i) d\mathbf{z} \\ &= \mathbb{E}_{p(\mathbf{Z}_i | \mathbf{Y}_i, \mathbf{X}_i)} \left[ \frac{p(\mathbf{z} | c_i = c, \mathbf{X}_i) p(c, \mathbf{X}_i)}{p(\mathbf{z}, \mathbf{X}_i)} \right] \\ &= \frac{(\mathbf{Z}_i, c_i) \perp\!\!\!\perp \mathbf{X}_i}{\mathbf{Z}_i, c_i \perp\!\!\!\perp \mathbf{X}_i} \mathbb{E}_{p(\mathbf{Z}_i | \mathbf{Y}_i, \mathbf{X}_i)} \left[ \frac{p(\mathbf{z} | c_i = c) p(c)}{p(\mathbf{z})} \right] \\ &\approx \frac{1}{L} \sum_{l=1}^L \frac{\hat{p}(\mathbf{z}^{(l)} | c_i = c) \hat{\pi}_c}{\sum_{c'=1}^K \hat{p}(\mathbf{z}^{(l)} | c_i = c') \hat{\pi}_{c'}} \triangleq \hat{p}(c | \mathbf{Y}_i, \mathbf{X}_i). \end{aligned}$$

372 So we can obtain the predicted category of cell  $i$  by computing the MAP estimator  $\hat{c}_i = \arg \max_c \hat{p}(c | \mathbf{Y}_i, \mathbf{X}_i)$ .

373 3.  $\tilde{w}_i | \mathbf{Y}_i, \mathbf{X}_i$

374 Note that

$$375 \quad p(w, c | \mathbf{Y}_i, \mathbf{X}_i) \approx \frac{1}{L} \sum_{l=1}^L \frac{\hat{p}(\mathbf{z}^{(l)} | c_i = c, w_i = w) \hat{\pi}_c}{\sum_{c'=1}^K \hat{p}(\mathbf{z}^{(l)} | c_i = c') \hat{\pi}_{c'}} \triangleq \hat{p}(w, c | \mathbf{Y}_i, \mathbf{X}_i).$$

376 Let  $\boldsymbol{\mu}_{\tilde{w}_i}$  and  $\Sigma_{\tilde{w}_i}$  denote the mean and the covariance matrix for the estimated distribution of  $\tilde{w}_i | \mathbf{Y}_i, \mathbf{X}_i$ . We have

$$377 \quad \begin{aligned} \boldsymbol{\mu}_{\tilde{w}_i} &= \sum_{c=1}^K \int [w a_c + (1-w) b_c] \hat{p}(w, c | \mathbf{Y}_i, \mathbf{X}_i) dw \\ &\approx \frac{1}{M} \sum_{c=1}^K \sum_{j=1}^M \left[ \frac{j}{M} a_c + \left(1 - \frac{j}{M}\right) b_c \right] \frac{1}{L} \sum_{l=1}^L \frac{\hat{p}(\mathbf{z}^{(l)} | c_i = c, w_i = \frac{j}{M}) \hat{\pi}_c}{\sum_{c'=1}^K \hat{p}(\mathbf{z}^{(l)} | c_i = c') \hat{\pi}_{c'}} \\ 378 \quad \text{diag}(\Sigma_{\tilde{w}_i}) &\approx \frac{1}{M} \sum_{c=1}^K \sum_{j=1}^M \left[ \frac{j}{M} a_c + \left(1 - \frac{j}{M}\right) b_c \right]^2 \frac{1}{L} \sum_{l=1}^L \frac{\hat{p}(\mathbf{z}^{(l)} | c_i = c, w_i = \frac{j}{M}) \hat{\pi}_c}{\sum_{c'=1}^K \hat{p}(\mathbf{z}^{(l)} | c_i = c') \hat{\pi}_{c'}} - \boldsymbol{\mu}_{\tilde{w}_i}^2. \end{aligned}$$

380 Here, as the integration over  $w$  is intractable, we approximate the integral by the rectangular rule with equally spaced  $M$   
381 points in  $[0, 1]$ .

382 **S2.3. Edge Score.** Below, seven kinds of edge scores are presented.

383 1. Edge Score Based on Posterior Mean of  $c$ .

384 The first edge score is based on the posterior mean  $\frac{1}{N} \sum_{i=1}^N p(c_i | \mathbf{Y}_i, \mathbf{X}_i)$ . Let

$$385 \quad \mathcal{C}_{j_1 j_2} = \{i : \mathbb{P}(c_i \in \{C_{j_1 j_1}, C_{j_1 j_2}, C_{j_2 j_2}\} | \mathbf{Y}_i, \mathbf{X}_i) > t_1\}$$

386 where  $t_1 = 0.5$  is the threshold parameter. We are considering  $\mathcal{C}_{j_1 j_2}$  instead of the full data set because not every cell  
387 contributes to the existence of the edge  $e_{j_1 j_2}$ . Within the set  $\mathcal{C}_{j_1 j_2}$ , proportion of cells at the edge  $e_{j_1 j_2}$  gives us one score

$$388 \quad \mathbb{E} \left[ \frac{1}{|\mathcal{C}_{j_1 j_2}|} \sum_{i \in \mathcal{C}_{j_1 j_2}} \mathbb{1}_{\{c_i = C_{j_1 j_2}\}} \right] \approx \frac{1}{|\hat{\mathcal{C}}_{j_1 j_2}|} \sum_{i \in \hat{\mathcal{C}}_{j_1 j_2}} \frac{\hat{p}(C_{j_1 j_2} | \mathbf{Y}_i, \mathbf{X}_i)}{\hat{p}(C_{j_1 j_1} | \mathbf{Y}_i, \mathbf{X}_i) + \hat{p}(C_{j_1 j_2} | \mathbf{Y}_i, \mathbf{X}_i) + \hat{p}(C_{j_2 j_2} | \mathbf{Y}_i, \mathbf{X}_i)},$$

389 which is denoted by  $s_{j_1 j_2}^{\text{mean}}$ , where

$$390 \quad \hat{\mathcal{C}}_{j_1 j_2} = \{i : \hat{p}(C_{j_1 j_1} | \mathbf{Y}_i, \mathbf{X}_i) + \hat{p}(C_{j_1 j_2} | \mathbf{Y}_i, \mathbf{X}_i) + \hat{p}(C_{j_2 j_2} | \mathbf{Y}_i, \mathbf{X}_i) > t_1\}.$$

391 2. Edge Score Based on Modified Posterior Mean of  $c$ .

392 A modified version of  $s_{j_1 j_2}^{\text{mean}}$  is given by

$$393 \quad s_{j_1 j_2}^{\text{modified\_mean}} = \frac{\sum_{i \in \hat{\mathcal{C}}_{j_1 j_2}} \hat{p}(C_{j_1 j_2} | \mathbf{Y}_i, \mathbf{X}_i)}{\sum_{i \in \hat{\mathcal{C}}_{j_1 j_2}} [\hat{p}(C_{j_1 j_1} | \mathbf{Y}_i, \mathbf{X}_i) + \hat{p}(C_{j_1 j_2} | \mathbf{Y}_i, \mathbf{X}_i) + \hat{p}(C_{j_2 j_2} | \mathbf{Y}_i, \mathbf{X}_i)]}$$

394 where an first-order approximation  $\mathbb{E} \left( \frac{X}{Y} \right) \approx \frac{\mathbb{E}X}{\mathbb{E}Y}$  is used.

395 3. Edge Score Based on Raw MAP of  $c$ .

396 Alternatively, we can also compute the edge score based on the MAP of  $c_i$ 's,

$$397 \quad \tilde{c}_i = \arg \max_{c \in \{1, 2, \dots, K\}} \hat{p}(c_i = c | \mathbf{Y}_i, \mathbf{X}_i).$$

398 Then an edge score can be defined as the proportion of the number of cells at the edge  $e_{j_1 j_2}$  to the number of cells at the  
399 vertices  $j_1, j_2$  or the edge  $e_{j_1 j_2}$ :

$$400 \quad s_{j_1 j_2}^{\text{raw\_map}} = \frac{|\{i : \tilde{c}_i = C_{j_1 j_2}\}|}{|\{i : \tilde{c}_i \in \{C_{j_1 j_2}, j_1 \neq j_2\}\}|}.$$

4. Edge Score Based on MAP of  $c$ .

We can also define the edge score of  $c_i$  as the proportion of the number of cells at the edge  $e_{j_1 j_2}$  to the number of cells at the vertices  $j_1, j_2$  or the edge  $e_{j_1 j_2}$ :

$$s_{j_1 j_2}^{\text{map}} = \frac{|\{i : \tilde{c}_i = C_{j_1 j_2}\}|}{|\{i : \tilde{c}_i \in \{C_{j_1 j_1}, C_{j_1 j_2}, C_{j_2 j_2}\}\}|}.$$

5. Edge Score Based on Modified MAP of  $c$ .

Since the assigned categories by MAP of  $c$  are generally at the edges instead of the nodes, the estimated denominator will be smaller than the real one, and  $s_{j_1 j_2}^{\text{map}}$  will be large when the number of cells in nodes  $j_1$  and  $j_2$  is small. To address this problem, we further modify the denominator to get the modified MAP edge score:

$$s_{j_1 j_2}^{\text{modified\_map}} = \frac{|\{i : \tilde{c}_i = C_{j_1 j_2}\}|}{|\{i : e_{j_1}^\top \boldsymbol{\mu}_{\tilde{w}_i} > 0.5 \text{ or } e_{j_2}^\top \boldsymbol{\mu}_{\tilde{w}_i} > 0.5\}|}.$$

6. Edge Score Based on MAP of  $\mu_{\tilde{w}_i}$ .

Sometimes the large number of cell type  $k$  will boost the dimension of  $\hat{p}(c_i = c | \mathbf{Y}_i, \mathbf{X}_i)$  to  $\frac{k(k-1)}{2}$ , which may cause the edge score inference sensitive to  $\hat{p}(c_i = c | \mathbf{Y}_i, \mathbf{X}_i)$  estimation. Therefore, in the dataset containing lots of cell types, we can compute the edge score from  $\mu_{\tilde{w}_i}$ , which only has  $k$  dimension and also contains the position information. First, define a set  $MAP(j_1, j_2)$  containing the cells that have the largest  $\mu_{\tilde{w}_i}$  value in the vertices  $j_1, j_2$  dimension.

$$MAP(j_1, j_2) := \{i : \arg \max_j e_j^T \mu_{\tilde{w}_i} = (j_1 \text{ or } j_2)\}$$

Then, the edge score can be defined by the proportion of the number of cells that lie in the middle of two vertices to the number of cells that have the largest  $\mu_{\tilde{w}_i}$  value in these vertices' dimensions:

$$s_{j_1, j_2}^{\text{w\_base}} = \frac{|\{i : \text{abs}(e_{j_1}^T \mu_{\tilde{w}_i} - e_{j_2}^T \mu_{\tilde{w}_i}) < 0.1, i \in MAP(j_1, j_2)\}|}{|MAP(j_1, j_2)|}$$

7. Edge Score Based on Modified MAP of  $\mu_{\tilde{w}_i}$ .

When there exists some unbalanced cell type categories,  $MAP(j_1, j_2)$  that only care about the 1st rank dimension of  $\mu_{\tilde{w}_i}$  may cause the edge score unexpectedly small when one of the vertices has a large number of cells while the other only has a small number. Then we can modified  $MAP(j_1, j_2)$  to consider the largest 2 dimension of  $\mu_{\tilde{w}_i}$ .

$$MAP2(j_1, j_2) := \{i : \arg \max_{j' \subset [k], |j'|=2} \sum_{j \in j'} e_j^T \mu_{\tilde{w}_i}, j' = \{j_1, j_2\}\}$$

Then, the edge score can be defined by the proportion of the number of cells that lie in the middle of the two vertices  $j_1, j_2$  to the number of cells that belong to  $MAP2(j_1, j_2)$ :

$$s_{j_1, j_2} = \frac{|\{i : \max_j e_j^T \mu_{\tilde{w}_i} < 0.55 \cdot (e_{j_1} + e_{j_2})^T \mu_{\tilde{w}_i}, i \in MAP2(j_1, j_2)\}|}{|MAP2(j_1, j_2)|}$$

By default, the package uses edge scores based on the modified MAP of  $c$ .

## S2.4. Proof of Proposition 1.

*Proof.* Note that

$$\begin{aligned} \mathbb{E}_{\hat{p}(\tilde{w}_i | \mathbf{Y}_i, \mathbf{X}_i)} \|\tilde{w}_i - \mathbf{w}\|_2^2 &= \mathbb{E}_{\hat{p}(\tilde{w}_i | \mathbf{Y}_i, \mathbf{X}_i)} (\tilde{w}_i^\top \tilde{w}_i) - 2\mathbf{w}^\top \tilde{w}_i + \mathbf{w}^\top \mathbf{w} \\ \mathbb{E}_{\hat{p}(\tilde{w}_i | \mathbf{Y}_i, \mathbf{X}_i)} \|\tilde{w}_i - \mathbf{w}\|_2^2 &= (\mathbb{E}_{\hat{p}(\tilde{w}_i | \mathbf{Y}_i, \mathbf{X}_i)} \tilde{w}_i)^\top \mathbb{E}_{\hat{p}(\tilde{w}_i | \mathbf{Y}_i, \mathbf{X}_i)} \tilde{w}_i - 2\mathbf{w}^\top \tilde{w}_i + \mathbf{w}^\top \mathbf{w}, \end{aligned}$$

we have

$$\begin{aligned} \mathbb{E}_{\hat{p}(\tilde{w}_i | \mathbf{Y}_i, \mathbf{X}_i)} \|\tilde{w}_i - \mathbf{w}\|_2^2 &= \|\mathbb{E}_{\hat{p}(\tilde{w}_i | \mathbf{Y}_i, \mathbf{X}_i)} \tilde{w}_i - \mathbf{w}\|_2^2 \\ &\quad - (\mathbb{E}_{\hat{p}(\tilde{w}_i | \mathbf{Y}_i, \mathbf{X}_i)} \tilde{w}_i)^\top \mathbb{E}_{\hat{p}(\tilde{w}_i | \mathbf{Y}_i, \mathbf{X}_i)} \tilde{w}_i + \mathbb{E}_{\hat{p}(\tilde{w}_i | \mathbf{Y}_i, \mathbf{X}_i)} (\tilde{w}_i^\top \tilde{w}_i), \end{aligned}$$

where the last two terms do not involve  $\mathbf{w}$ . So, minimizing  $\mathbb{E}_{\hat{p}(\tilde{w}_i | \mathbf{Y}_i, \mathbf{X}_i)} \|\tilde{w}_i - \mathbf{w}\|_2^2$  is equivalent to minimizing  $\|\mathbb{E}_{\hat{p}(\tilde{w}_i | \mathbf{Y}_i, \mathbf{X}_i)} \tilde{w}_i - \mathbf{w}\|_2^2$  over  $\mathbf{w}$ . The optimization problem (10) is thus equivalent to the optimization problem (11).

We divide the problem into two cases: finding the best projection onto edges and the best projection onto vertices.

432 (1) For projection onto edges, the original problem becomes

$$433 \quad \min_{(j_1, j_2) \in \mathcal{E}(\widehat{\mathcal{B}})} (\mathbf{w}_{j_1} - \mu_{j_1})^2 + (\mathbf{w}_{j_2} - \mu_{j_2})^2 - (\mu_{j_1}^2 + \mu_{j_2}^2)$$

$$434 \quad \text{s.t. } \|\mathbf{w}\|_1 = w_{j_1} + w_{j_2} = 1, \quad \mathbf{w} \succeq \mathbf{0}_k.$$

435 Notice that for any edge  $(j_1, j_2) \in \mathcal{E}(\widehat{\mathcal{B}})$ , the objective function is quadratic in  $w_{j_1}$  (and  $w_{j_2}$ ) since

$$436 \quad (w_{j_1} - \mu_{j_1})^2 + (w_{j_2} - \mu_{j_2})^2 - (\mu_{j_1}^2 + \mu_{j_2}^2)$$

$$437 \quad = (w_{j_1} - \mu_{j_1})^2 + (1 - w_{j_1} - \mu_{j_2})^2 - (\mu_{j_1}^2 + \mu_{j_2}^2)$$

$$438 \quad = 2 \left( w_{j_1} - \frac{1 + \mu_{j_1} - \mu_{j_2}}{2} \right)^2 + \frac{(1 - \mu_{j_1} - \mu_{j_2})^2}{2}$$

$$439 \quad \geq \frac{(1 - \mu_{j_1} - \mu_{j_2})^2}{2},$$

440 with equality holds if and only if  $w_{j_1} = \mu_{j_1} + \frac{1 - \mu_{j_1} - \mu_{j_2}}{2}$ . That is, for each edge  $(j_1, j_2) \in \mathcal{E}(\widehat{\mathcal{B}})$ , the solution

$$441 \quad w_{k'} = \begin{cases} \mu_{j_1} + \frac{1 - \mu_{j_1} - \mu_{j_2}}{2}, & k' = j_1 \\ \mu_{j_2} + \frac{1 - \mu_{j_1} - \mu_{j_2}}{2}, & k' = j_2 \\ 0, & \text{otherwise,} \end{cases}$$

442 achieves minimum projection error

$$443 \quad \|\mathbf{w} - \mu_{\widehat{\mathbf{w}}}\|_2^2 = \sum_{k' \neq j_1, j_2} \mu_{k'}^2 + \frac{(1 - \mu_{j_1} - \mu_{j_2})^2}{2}.$$

444 If we want edge  $(j_1, j_2)$  to have the smallest projection error, then for any other edge  $(m_1, m_2) \in \mathcal{E}(\widehat{\mathcal{B}})$ , we need to have

$$445 \quad \|\text{proj}_{(j_1, j_2)}(\mu_{\widehat{\mathbf{w}}}) - \mu_{\widehat{\mathbf{w}}}\|_2^2 \leq \|\text{proj}_{(m_1, m_2)}(\mu_{\widehat{\mathbf{w}}}) - \mu_{\widehat{\mathbf{w}}}\|_2^2$$

$$446 \quad \iff \sum_{k' \neq j_1, j_2} \mu_{k'}^2 + \frac{(1 - \mu_{j_1} - \mu_{j_2})^2}{2} \leq \sum_{k' \neq m_1, m_2} \mu_{k'}^2 + \frac{(1 - \mu_{m_1} - \mu_{m_2})^2}{2} \quad [16]$$

$$447 \quad \iff (\mu_{j_1} - \mu_{j_2})^2 + 2(\mu_{j_1} + \mu_{j_2}) \geq (\mu_{m_1} - \mu_{m_2})^2 + 2(\mu_{m_1} + \mu_{m_2}).$$

448 Thus, the best-projected edge can be obtained by computing

$$449 \quad (j_1^*, j_2^*) = \arg \max_{(j_1, j_2) \in \mathcal{E}(\widehat{\mathcal{B}})} (\mu_{j_1} - \mu_{j_2})^2 + 2(\mu_{j_1} + \mu_{j_2}).$$

450 (2) For the vertex  $j \in \mathcal{N}(\widehat{\mathcal{B}})$ , the problem becomes

$$451 \quad \min_{\mathbf{w}} (w_j - \mu_j)^2 \quad \text{s.t. } \|\mathbf{w}\|_1 = w_j = 1, \quad \mathbf{w} \succeq \mathbf{0}_k,$$

452 and the solution is  $\text{proj}_j(\mu_{\widehat{\mathbf{w}}}) = \mathbf{e}_j$ . The projection error for the vertex  $i$  is given by

$$453 \quad \|\text{proj}_j(\mu_{\widehat{\mathbf{w}}}) - \mu_{\widehat{\mathbf{w}}}\|_2^2 = \sum_{k' \neq j} \mu_{k'}^2 + (\mu_j - 1)^2.$$

454 Then  $j_3^* \in \arg \min_{j \in \mathcal{N}(\widehat{\mathcal{B}})} \|\text{proj}_j(\mu_{\widehat{\mathbf{w}}}) - \mu_{\widehat{\mathbf{w}}}\|_2^2$  gives the best projection onto vertices.

455 (3) Consider the following case,

$$456 \quad \|\text{proj}_{j_3^*}(\mu_{\widehat{\mathbf{w}}}) - \mu_{\widehat{\mathbf{w}}}\|_2^2 \leq \|\text{proj}_{(j_1^*, j_2^*)}(\mu_{\widehat{\mathbf{w}}}) - \mu_{\widehat{\mathbf{w}}}\|_2^2$$

$$457 \quad \iff \sum_{k' \neq j_3^*} \mu_{k'}^2 + (1 - \mu_{j_3^*})^2 \leq \sum_{k' \neq j_1^*, j_2^*} \mu_{k'}^2 + \frac{(1 - \mu_{j_1^*} - \mu_{j_2^*})^2}{2}$$

$$458 \quad \iff \mu_{j_1^*}^2 + \mu_{j_2^*}^2 + (1 - \mu_{j_3^*})^2 \leq \mu_{j_3^*}^2 + \frac{(1 - \mu_{j_1^*} - \mu_{j_2^*})^2}{2},$$

$$459 \quad \iff 4\mu_{j_3^*} - 1 \geq (\mu_{j_1^*} - \mu_{j_2^*})^2 + 2(\mu_{j_1^*} + \mu_{j_2^*})$$

$$460 \quad \iff (\mu_{j_3^*} - \mu_{j_3^*})^2 + 2(\mu_{j_3^*} + \mu_{j_3^*}) - 1 \geq (\mu_{j_1^*} - \mu_{j_2^*})^2 + 2(\mu_{j_1^*} + \mu_{j_2^*}).$$

461 Define  $\mathcal{EN}(\mathcal{G}) = \mathcal{E}(\mathcal{G}) \cup \{(j, j) : j \in \mathcal{N}(\mathcal{G})\}$ , then

$$462 \quad \arg \max_{(j_1, j_2) \in \mathcal{EN}(\widehat{\mathcal{B}})} (\mu_{j_1} - \mu_{j_2})^2 + 2(\mu_{j_1} + \mu_{j_2}) - \mathbb{1}_{\{j_1=j_2\}}$$

463 gives solutions to both optimization problems. □

### 464 S3. Benchmarking

465 **S3.1. Benchmarking Datasets.** Following the settings in (1), which provided a comprehensive overview and guideline for  
 466 TI methods, we evaluate VITAE’s performance in recovering six types of trajectory topologies, as shown in Fig. S1. Our  
 467 benchmarking datasets include 10 real scRNA-seq data and 15 synthetic datasets, summarized in Table S1.

468 The datasets from real scRNA-seq studies include 9 datasets from (1). Among them, 5 datasets have “gold standard” labels  
 469 according to (1) and are included to cover all trajectory types with at least 200 cells. As most “gold standard” datasets are  
 470 small, we also include 4 extra datasets (*dentate*, *fibroblast*, *planaria\_muscle*, *planaria\_full*) with “silver standard” labels. The  
 471 datasets and labels are all extracted from the Dyno platform (1), except for the dataset *dentate*, whose cells are mislabeled, and  
 472 we directly extract the labels from the GEO database (accession number: GSE95315). For each dataset, the Dyno platform also  
 473 provides its reference trajectory backbone and cell positions, from which we calculate reference pseudotime by our definition. In  
 474 addition, to evaluate TI methods on a dataset with disconnected states, we create the dataset *immune* by combining purified  
 475 10085 B cells, 8385 CD56 NK cells, and 2612 CD14 Monocytes cells from (26).

476 The drawback of using real datasets for benchmarking is that only discrete cell labels are available, though the cells are  
 477 experiencing continuous transitions. In other words, the true cell positions and ordering along the trajectory are only known at  
 478 a low resolution. To better evaluate VITAE’s performance in estimating the cell positions and pseudotime, we also include  
 479 synthetic datasets for evaluation. We consider two different simulation approaches. One simulator we use is *dyngen* (27), a  
 480 multi-modal simulation engine for studying dynamic cellular processes at single-cell resolution. *dyngen* is also used in (1) and  
 481 provides a delicate way to generate scRNA-seq data starting from gene regulation and transcriptional factors. However, it is  
 482 limited to generating only a few hundred genes. Thus, we only generate 1000 genes for each *dyngen* dataset and treat the  
 483 *dyngen* datasets as non-UMI data as the generated counts are typically large.

484 We also generate four synthetic datasets from our own model framework. First, we train the model on a real data set to  
 485 obtain a decoder and  $\hat{U}$ . Then, we generate each  $\hat{\mathbf{w}}_i$  and  $\mathbf{Z}_i$  and the observed UMI counts  $\mathbf{Y}_i$  following the hierarchical models  
 486 (Eq. (3) and 6). Specifically, we treat the estimated decoder and  $\hat{U}$  as the true  $f_g(\cdot)$  and  $U$  and design a trajectory backbone by  
 487 connecting some edges between the vertices. In the four generated datasets, we do not include any confounding covariates  $\mathbf{X}_i$ .  
 488 We use real dataset *dentate* to generate synthetic datasets *linear* and *bifurcation*, and the real dataset *fibroblast* to generate  
 489 synthetic datasets *multifurcating* and *tree*.

490 **S3.2. Evaluation metrics.** We use five different scores to measure TI methods’ accuracy in recovering the true trajectory, cell  
 491 positions, and pseudotime. All these scores range from 0 to 1, and a larger value indicates better performance.

492 First, to measure the difference between an estimated trajectory backbone  $\hat{\mathcal{B}}$  and the true trajectory backbone  $\mathcal{B}$ , we compute  
 493 two scores: the GED score and the IM score, both of which evaluate the difference between two graphs and are invariant to the  
 494 permutation of vertices. The graph edit distance (GED) is defined as  $\text{GED}(\hat{\mathcal{B}}, \mathcal{B}) = \min_{e_1, \dots, e_j \in \mathcal{P}(\hat{\mathcal{B}}, \mathcal{B})} \sum_{i=1}^j c(e_i)$  where  $\mathcal{P}(\hat{\mathcal{B}}, \mathcal{B})$   
 495 denotes the set of edit paths transforming graph  $\hat{\mathcal{B}}$  to the graph  $\mathcal{B}$  and  $c(e_i) \geq 0$  is the cost of an operation  $e_i$ . In other words,  
 496 GED is a symmetric distance measure that quantifies the minimum number of graph edit operations needed to transform  $\hat{\mathcal{B}}$   
 497 into an isomorphic graph of  $\mathcal{B}$ . We then standardize the GED as

$$498 \quad s_{\text{GED}} = 1 - \min \left\{ \text{GED}(\hat{\mathcal{B}}, \mathcal{B})/k, 1 \right\},$$

499 so that it ranges between 0 and 1. Besides, following (1), we also compute the IM score based on the Ipsen-Mikhailov  
 500 distance (29),  $\text{IM}(\mathcal{B}, \hat{\mathcal{B}})$ , between the estimated and reference trajectory backbone. The IM distance is symmetric, measures  
 501 the dissimilarity of adjacency matrices’ spectra of the two graphs, and is bounded between 0 and 1. Our IM score is further  
 502 defined as  $s_{\text{IM}} = 1 - \text{IM}(\mathcal{B}, \hat{\mathcal{B}})$ .

503 Next, we measure the error of the estimated cell position  $\hat{\mathbf{w}}_i$  for each cell  $i$ . We use two scores, the adjusted rand index  
 504 (ARI) and the generalized rand index (GRI), both of which are invariant under the permutation of vertices and allow an  
 505 unequal number of vertices. The ARI (30) is a commonly used symmetric measure for the similarity between two clustering  
 506 results. Following (1), the cells are assigned to their nearest discrete state based on their estimated  $\hat{\mathbf{w}}_i$  and true values of  $\tilde{\mathbf{w}}_i$ ,  
 507 and the two groupings are compared using ARI. Since such discretization only compares the accuracy of  $\hat{\mathbf{w}}_i$  at a low resolution,  
 508 we also define a GRI score that directly compares  $\hat{\mathbf{w}}_i$  with  $\tilde{\mathbf{w}}_i$ , which is an extension of the rand index (RI). For any two pair of  
 509 cells  $i_1$  and  $i_2$ , we define its “true” similarity as  $\rho_{i_1 i_2} = \langle \sqrt{\tilde{\mathbf{w}}_{i_1}}, \sqrt{\tilde{\mathbf{w}}_{i_2}} \rangle$  where  $\sqrt{\mathbf{v}} = (\sqrt{v_1}, \dots, \sqrt{v_k})$  for a vector  $\mathbf{v}$  of length  $k$ .  
 510 The square-root operation is to guarantee that  $\rho_{i_1 i_2} = 1$  if and only if  $\tilde{\mathbf{w}}_{i_1} = \tilde{\mathbf{w}}_{i_2}$ . Similarly, we define the estimated similarity  
 511 between the two cells as  $\hat{\rho}_{i_1 i_2} = \langle \sqrt{\hat{\mathbf{w}}_{i_1}}, \sqrt{\hat{\mathbf{w}}_{i_2}} \rangle$ . Then to compare the similarities between the estimated and reference cell  
 512 positions on the trajectory backbone, the GRI is defined as

$$513 \quad \text{GRI} = \sum_{i_1=1}^{n-1} \sum_{i_2=i_1+1}^n (1 - |\rho_{i_1 i_2} - \hat{\rho}_{i_1 i_2}|) / \binom{2}{n}.$$

514 The rand index is a special case of GRI when all cells are positioned exactly on vertices.

515 Finally, we also measure the similarity between the reference and estimated pseudotime of cells along the trajectory, which is  
 516 simply the Pearson correlation between the estimated pseudotime  $\hat{T}_i$  and the reference pseudotime  $T_i$  across cells. Specifically,  
 517 we define the PDT score ranging between 0 and 1 as  $s_{\text{PDT}} = \left( \text{Cor}(\mathbf{T}, \hat{\mathbf{T}}) + 1 \right) / 2$ .

518 **S3.3. DE analysis.** We compare VITAE with tradeSeq (15) and PseudotimeDE (16) on differentially expressed testing. Similar  
519 to (15), we use package `dyntoy` from `dynverse` toolbox (27) to generate linear trajectories. We perform 10 simulations. For each  
520 simulation, we generate a synthetic scRNA-seq dataset with 5000 genes and varying numbers of cells in  $n \in \{250, 500, 750, 1000\}$ .  
521 We set the differentially expressed rate to be 0.2, so that 20% of the genes are active. Then, we apply both Slingshot and  
522 VITAE on the simulated datasets to infer trajectory and estimate the pseudotime for all cells. We next apply DE test methods  
523 VITAE, tradeSeq, and PseudotimeDE on the estimated pseudotime with BH procedure and significance level 5%. We provide  
524 the details of the configuration below.

525 For Slingshot, we follow the usual preprocessing procedure to use all the genes for dimension reduction with PCA and  
526 clustering the cells with the default parameters of Seurat. Then we extract the principle components, cluster assignment for all  
527 cells, and the initial cluster as the input to Slingshot. For tradeSeq, we follow their workflow to call `fitGAM` function with  
528 `nknots=4` and use `associationTest` function with `contrastType='consecutive'`, which returns the adjusted  $p$ -values for all  
529 genes. For PseudotimeDE, we use 50 random subsampled raw count datasets of size  $0.8n$ , and apply Slingshot to estimate  
530 the pseudotime for each of the datasets. Then, we call `runPseudotimeDE` function with parameter `model = 'nb'` and 10 CPU  
531 cores to obtain the adjusted  $p$ -values. We evaluate the three DE test methods using pseudotime estimated from trajectory  
532 inference methods Slingshot and VITAE. We did not explore PseudotimeDE based on VITAE's estimated pseudotime due to  
533 the computational cost. Therefore, our comparison consists of five combinations of TI and DE methods: Slingshot+VITAE,  
534 Slingshot+tradeSeq, Slingshot+PseudotimeDE, VITAE+VITAE, and VITAE+tradeSeq.

535 The results of the false discovery rate (FDR) and true positive rate (TPR) are shown in Fig. S2. The signals are strong in  
536 the synthetic data (the signal strength is not adjustable in `dyntoy`), so all methods have powers of almost 1. In terms of FDR  
537 control, we have better control than tradeSeq. Compared with PseudotimeDE, our method has similar FDR control but is less  
538 computationally expensive. In terms of computational complexity, for a dataset with 1000 cells and 5000 genes, it took about  
539 1 min, 2 min, and 2 h for VITAE+VITAE, Slingshot+tradeSeq, and Slingshot+PseudotimeDE to perform both trajectory  
540 inference and testing.

## 541 S4. Real data

542 **S4.1. Experiments on mouse brain datasets.** To adjust for the confounding batch effect, we conducted 10 random trials on the  
543 joint analysis of two mouse neocortex datasets (Yuzwa and Ruan datasets), both with ( $\gamma = 1$ ) and without ( $\gamma = 0$ ) this loss,  
544 keeping all other hyperparameters at their default settings. As a stability measure of the estimated cell state centers in the  
545 latent space (denoted as  $\mathbf{U}$ ) and the estimated probabilities for a cell to choose a vertex/edge (denoted as  $\boldsymbol{\pi}$ ), we calculated  
546 the standard deviation of the pairwise distances between columns of  $\mathbf{U}$  (pairwise distances between cluster centers) and the  
547 standard deviation of each entry in  $\boldsymbol{\pi}$  across the 10 trials. In Fig. S3, we observe a substantial improvement in the stability of  
548 our estimates with the Jacobian regularizer.

## 549 S4.2. Experiments on the integration of three mouse brain datasets.

550 **Experiment setup.** To adjust for confounding batch effect, we include the dataset ID of each cell (whether it is from Di Bella's,  
551 Yuzwa's, or Ruan's dataset) as covariates. In addition, as the Apical Progenitors are experiencing cell division, we also adjust  
552 for the cell-cycle effects by adding the cell-cycle scores (`S.Score` and `G2M.Score`) provided by their own datasets. To perform  
553 the joint trajectory analysis, we apply `VITAE_Gauss` with the aforementioned covariates. We provide the annotated cell types  
554 to initialize the latent space after the pretraining step. For the model hyperparameters, we use 96 PCs, two hidden layers with  
555 units 48 and 24 for the encoder, a symmetric structure for the decoder, and 16 units for the latent space. Since the dataset is  
556 larger and more complicated, we retain more PCs as input and use a wider network structure. For pre-training and training,  
557 we set  $\gamma = 1$  for the Jacobian regularizers.

558 **Integration.** We further evaluated the integration capabilities of VIATE against other baseline methods, including Seurat  
559 CCA (31), Monocle 3 (32), scVI (33), and Scanorama (34). The latter two are among the leading integration methods that  
560 are suitable for downstream trajectory inference (35). The VIATE model setup is detailed in the preceding section. For  
561 Seurat CCA, we employed the `Seurat V3` R package and the `RunCCA` function. With `Monocle 3`, the `align_cds` function was  
562 utilized. For scVI, source information was incorporated using the `batch_key` parameter in the `scvi.model.SCVI` function.  
563 Lastly, Scanorama integration was achieved through the `scanorama.correct_scanpy` function. All integration processes are  
564 succeeded by the application of UMAP to derive a 2-dimensional embedding of the integrated dataset.

565 **Evaluation metric for integration.** We construct a mixing score based on the reduced UMAP space to measure the mixing effect  
566 of Day E18 Replicate, Day P1 Replicate, different days, and different cell types within the same day. We first calculate the  
567 KNN graph in UMAP space. Suppose we have  $n$  categories, and we want to quantify how these  $n$  categories are mixed. In the  
568  $k$  neighbors of a cell, assume there are  $x$  cells that have the same category as this cell; then, for each cell, the mixing metric  
569 can be calculated by

$$570 \text{ mixing score} = 1 - \frac{(x - \frac{k}{n})^+ + [k - \frac{k}{n} - (k - x)]^+}{2(1 - \frac{1}{n})k},$$

571 where  $(x - a)^+ = \max\{x - a, 0\}$ . The score reaches 1 when  $x \leq k/n$  and reaches 0 when  $x = k$ . The denominator  $2(1 - n^{-1})k$  is  
 572 used to normalize the score between 0 and 1. This score can also be converted to quantify the separation of different categories:

573 
$$\text{separation score} = 1 - \text{mixing score}.$$

574 Consequently, we use the mixing score to measure the mixing effect of Day E18 Replicate, Day P1 Replicate, and different  
 575 days. The separation score is applied to quantify the biological cell type conservation on different days.

576 **S4.3. Experiments on multi-omic human hematopoiesis datasets.** To adjust for confounding batch effect, we include the  
 577 sequencing technique of each cell (scRNA-seq or scATAC-seq) and also use MMD loss calculated within each annotated cell  
 578 type to remove effects from different sequencing techniques better. For the model hyperparameters, we use 128 PCs, two  
 579 hidden layers with units 32 and 16 for the encoder, a symmetric structure for the decoder, and 8 units for the latent space. For  
 580 pre-training, we set  $\gamma = \kappa = 0.6$  for the Jacobian and MMD regularizers. For training, we set  $\gamma = \kappa = 1$  for the Jacobian and  
 581 MMD regularizers.

582 We compare the trajectory results of VITAE to VIA (36) v0.1.96. Following the procedure in the original VIA paper, which  
 583 involves employing Seurat CCA for cross-modality data integration and subsequently applying VIA to the merged dataset  
 584 for trajectory analysis, we compare the visualization, inferred trajectory structure and pseudotime results between VIA and  
 585 VITAE. As shown in Figure S8c, the inferred trajectory of VIA is less interpretable compared to VITAE. This is partially  
 586 contributed by the Seurat’s CCA integration method, which tends to overcorrect the cell embeddings, particularly affecting cell  
 587 types with a limited number of cells, such as in the B cell lineage. Additionally, VIA fails to provide meaningful separations  
 588 between some cell types even though the annotated cell types are provided as the input. Moreover, even when two cell types  
 589 are visually distinct to each other, such as Baso.Eryth and Pre.B, VIA still incorrectly infers a connection between them.

590 **S5. Analysis of hyperparameters sensitivity and computational efficiency**

591 **S5.1. Hyperparameter  $\gamma$  for Jacobian regularization.** We conducted 10 random trials on the joint analysis of the two mouse  
 592 neocortex datasets (datasets from Yuzwa et. al (37) and Ruan et. al (38)), both with ( $\gamma = 1$ ) and without ( $\gamma = 0$ ) this loss,  
 593 keeping all other hyperparameters at their default settings. As a stability measure of the estimated cell state centers ( $\mathbf{U}$ ) in the  
 594 latent space, we calculated the standard deviation of the pairwise distances between columns of  $\mathbf{U}$  (pairwise distances between  
 595 cluster centers) across the 10 trials. We also measure the stability of the estimated prior probabilities for a cell to choose  
 596 each vertex/edge ( $\boldsymbol{\pi}$ ), by calculating the standard deviation of each entry of  $\boldsymbol{\pi}$  across the 10 trials. In Fig. S3, we observe a  
 597 substantial improvement in the stability of our estimates with the Jacobian regularizer.

598 **S5.2. Hyperparameter  $\alpha$  for the Soft penalty term.** We vary the value of the hyperparameter  $\alpha$  on the soft penalty term in the  
 599 joint analysis of the two mouse neocortex datasets (datasets from Yuzwa et. al (37) and Ruan et. al (38)), keeping all other  
 600 hyperparameters in their default values. Fig. S0.2 shows the embedding of cells and the inferred trajectory after training with  
 601 different  $\alpha$  values:  $\alpha = 0, 0.1, 0.15, 0.2, 0.5, 1$ . We observe that, with the soft penalty terms, VITAE is more accurate in learning  
 602 the shared trajectory of the two datasets. Its performance is relatively robust to minor adjustment of  $\alpha$ .

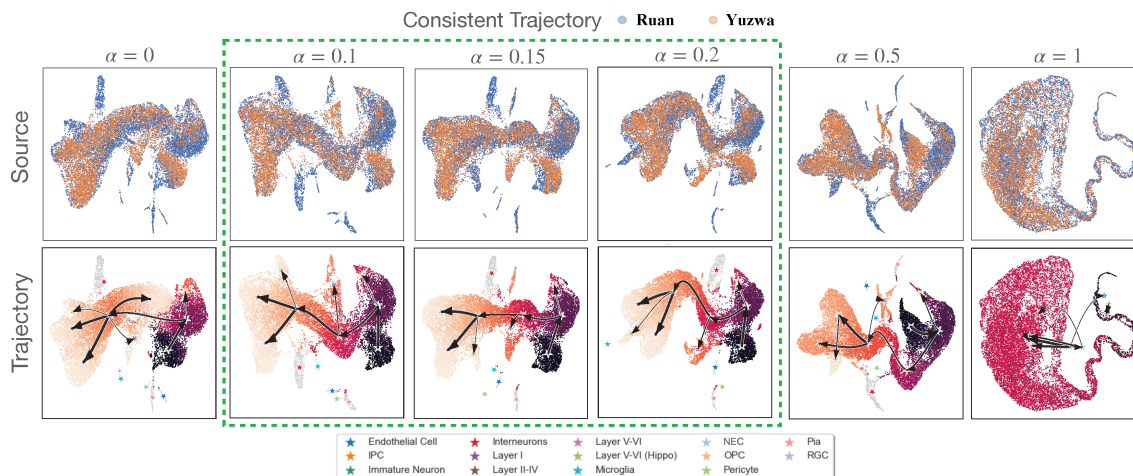


Fig S0.2. Sensitivity of VITAE on parameter  $\alpha$  of soft penalty term.

603 **S5.3. Hyperparameter  $\kappa$  for MMD regularization.** We investigate the effect of varying the MMD loss penalty  $\kappa$  in the joint  
 604 trajectory analysis of scATAC-seq and scRNA-seq hematopoiesis data where the MMD loss is used to encourage the merging of  
 605 two datasets. Fig. S0.3 shows the embedding of cells and the inferred trajectory after training with  $\kappa$  values ranging from 0 to  
 606 1.5, keeping all other hyperparameters unchanged. Without MMD loss, the cells from the scRNA-seq and the scATAC-seq  
 607 datasets cannot merge well. On the other hand, if  $\kappa$  is too large, even though the cells can merge well, the latent representations  
 608 are less dispersed, making it hard to infer the trajectory correctly. VITAE is relatively insensitive to  $\kappa$  if  $\kappa$  varies within a  
 609 certain range.

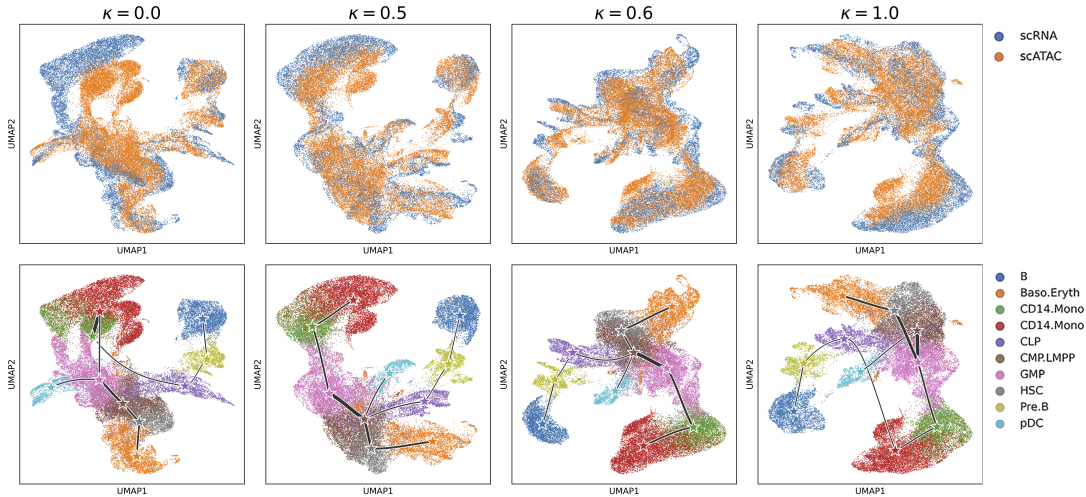


Fig S0.3. Sensitivity of VITAE on parameter  $\kappa$  of MMD loss.

610 **S5.4. Computation efficiency and memory usage.** We compare the computational efficiency of VITAE and alternative approaches  
 611 when performing joint trajectory analysis on multiple datasets. Because Slingshot takes the dimension-reduced object as the  
 612 input, it relies on data integration methods, such as Seurat's CCA, to adjust for potential batch effects. Therefore, it is more  
 613 reasonable to compare the time complexity of both integration and trajectory inference.

614 To this end, we compared four different approaches:

- 615 1. VITAE that adjusts for both the data source differences and cell cycle scores;
- 616 2. Monocle with data source correction but no cell cycle score adjustments (as it can not adjust for continuous cell cycle  
 617 scores);
- 618 3. Seurat that adjusts for both the data source differences and cell cycle scores to integrate the two datasets;
- 619 4. Seurat with data source correction but no cell cycle score adjustments.

620 The last two methods above provide an estimate of the shortest time for running Slingshot on the data integrated by Seurat.  
 621 To evaluate these methods on different sizes of data, we subsampled cells from the joint datasets from Yuzwa et. al (37) and  
 622 Ruan et. al (38), and generated a sequence of datasets of increasing cell numbers  $n = 1000, 2000, \dots, 15000, 16651$ . As shown  
 623 in Fig. S0.4, though VITAE is slightly slower than Monocle, it is faster than Seurat when adjusting for both batch effects and  
 624 cell cycle scores.

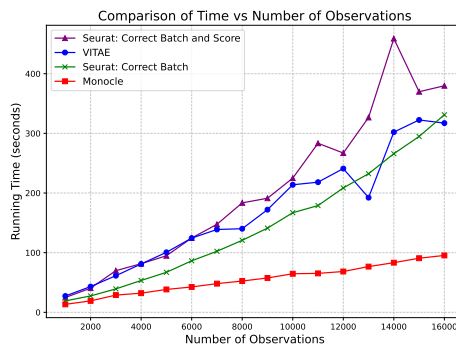


Fig S0.4. Computational time in the size of datasets for different methods.

625 We also record the memory usage of VITAE in real data analysis. The joint analysis of the two mouse neocortex data  
626 (Yuzwa et. al (37) and Ruan et. al (38)) is successfully executed on an 11th Gen Intel Core i7 CPU with 16 GB of RAM. The  
627 joint trajectory analysis of three datasets from Di Bella et. al (39), Yuzwa et. al (37), and Ruan et. al (38) can be performed  
628 on an Intel Xeon Gold CPU with 32 GB of RAM and a total running time of approximately 27 minutes.

**Table S1. Detailed summary of datasets used in experiments, including trajectory topology, number of cells  $N$ , number of genes  $G$ , number of cell types  $k$ , and sources of the dataset.**

Type	Name	Count Type	Topology	$N$	$G$	$k$	Source
real	aging	non-UMI	linear	873	2815	3	(40)
	human_embryos	non-UMI	linear	1289	8772	5	(41)
	germline	non-UMI	bifurcation	272	8772	7	(42)
	mesoderm	non-UMI	tree	504	8772	9	(43)
	cell_cycle	non-UMI	cycle	264	6812	3	(41)
	dentate	UMI	linear	3585	2182	5	(44)
	fibroblast	non-UMI	bifurcation	355	3379	7	(45)
	planaria_muscle	UMI	bifurcation	2338	4210	3	(46)
	planaria_full	UMI	tree	18837	4210	33	(46)
	immune	UMI	disconnected	21082	18750	3	(26)
synthetic	linear_1	non-UMI	linear	2000	991	4	dyngen
	linear_2	non-UMI	linear	2000	999	4	dyngen
	linear_3	non-UMI	linear	2000	1000	4	dyngen
	bifurcating_1	non-UMI	bifurcation	2000	997	7	dyngen
	bifurcating_2	non-UMI	bifurcation	2000	991	7	dyngen
	bifurcating_3	non-UMI	bifurcation	2000	1000	7	dyngen
	trifurcating_1	non-UMI	multifurcating	2000	969	10	dyngen
	trifurcating_2	non-UMI	multifurcating	2000	995	10	dyngen
	converging_1	non-UMI	bifurcation	2000	998	6	dyngen
	cycle_1	non-UMI	cycle	2000	1000	3	dyngen
	cycle_2	non-UMI	cycle	2000	999	3	dyngen
	cycle_3	non-UMI	cycle	2000	999	3	dyngen
	linear	UMI	linear	1900	1990	5	our model
	bifurcation	UMI	bifurcation	2000	606	6	our model
	multifurcating	UMI	multifurcating	2000	606	6	our model
tree	UMI	tree	2000	606	6	our model	

**Table S2. Number of cells on different embryonic days.**

Embryonic Day	E10.5	E11.5	E12.5	E13.5	E14.5	E15.5	E16.5	E17.5	E18.5
Ruan's Dataset	1172	0	2668	0	3742	1014	387	0	1278
Yuzwa's Dataset	0	1418	0	1137	0	2955	0	880	0

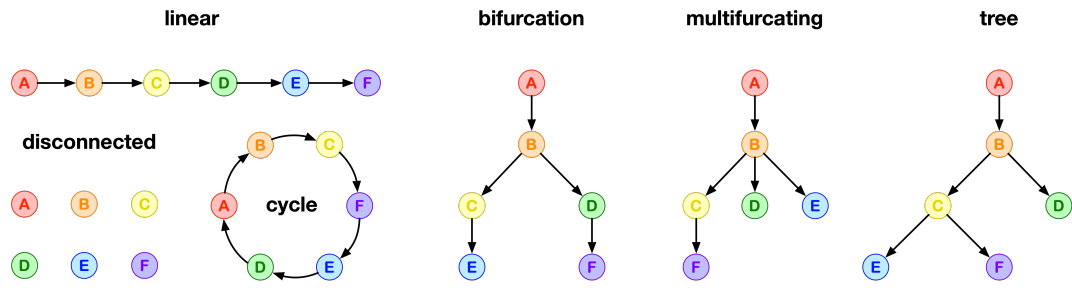
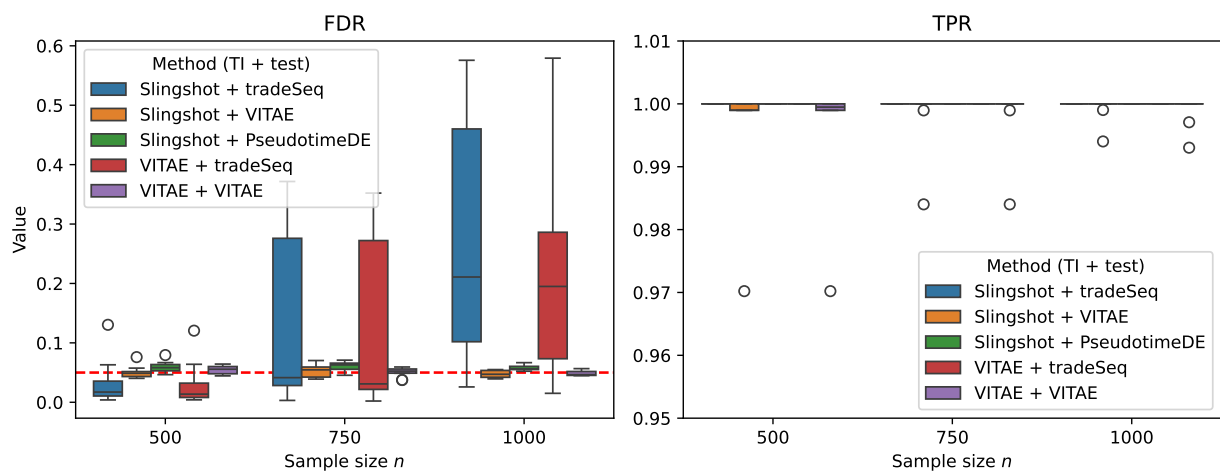
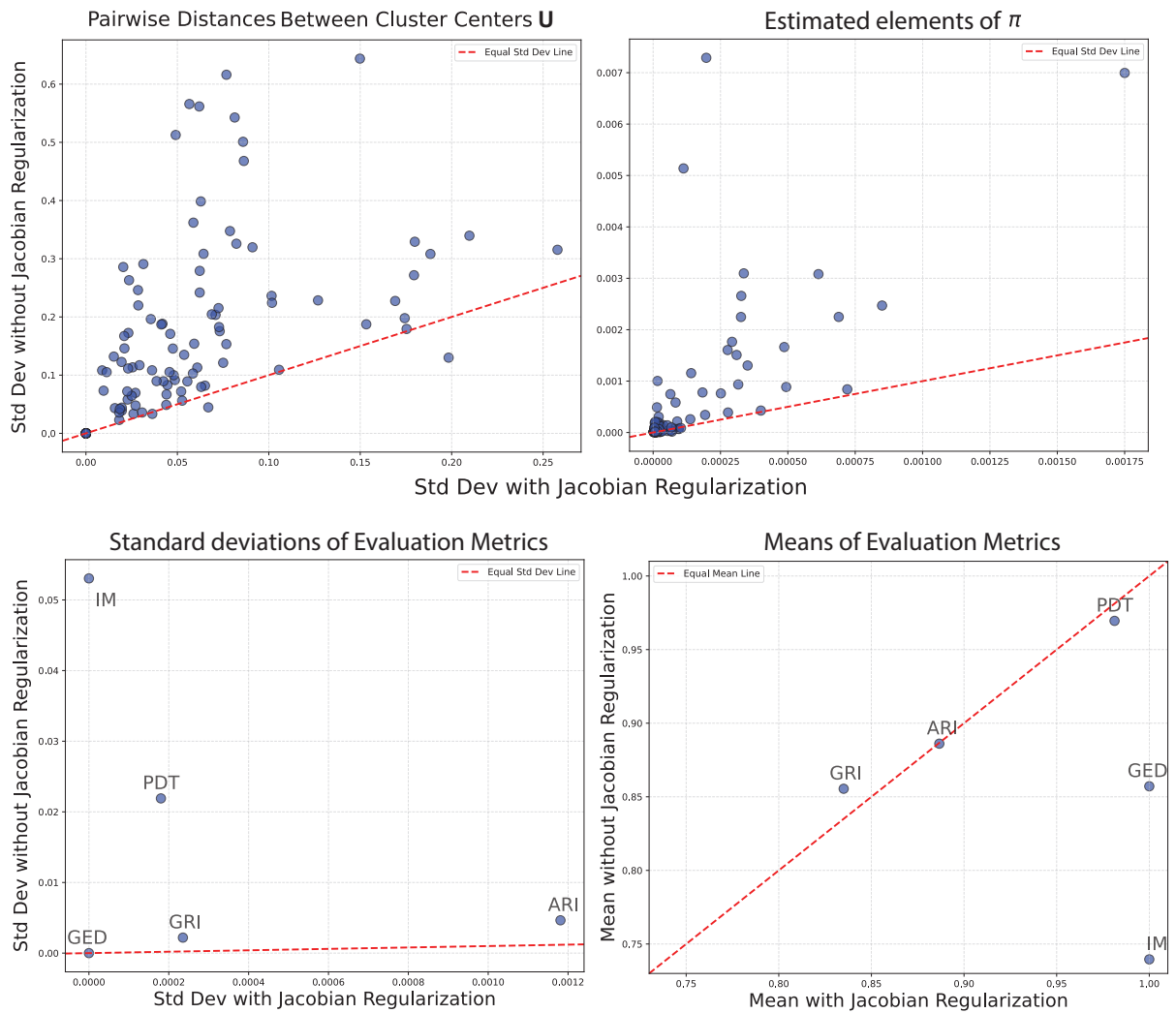


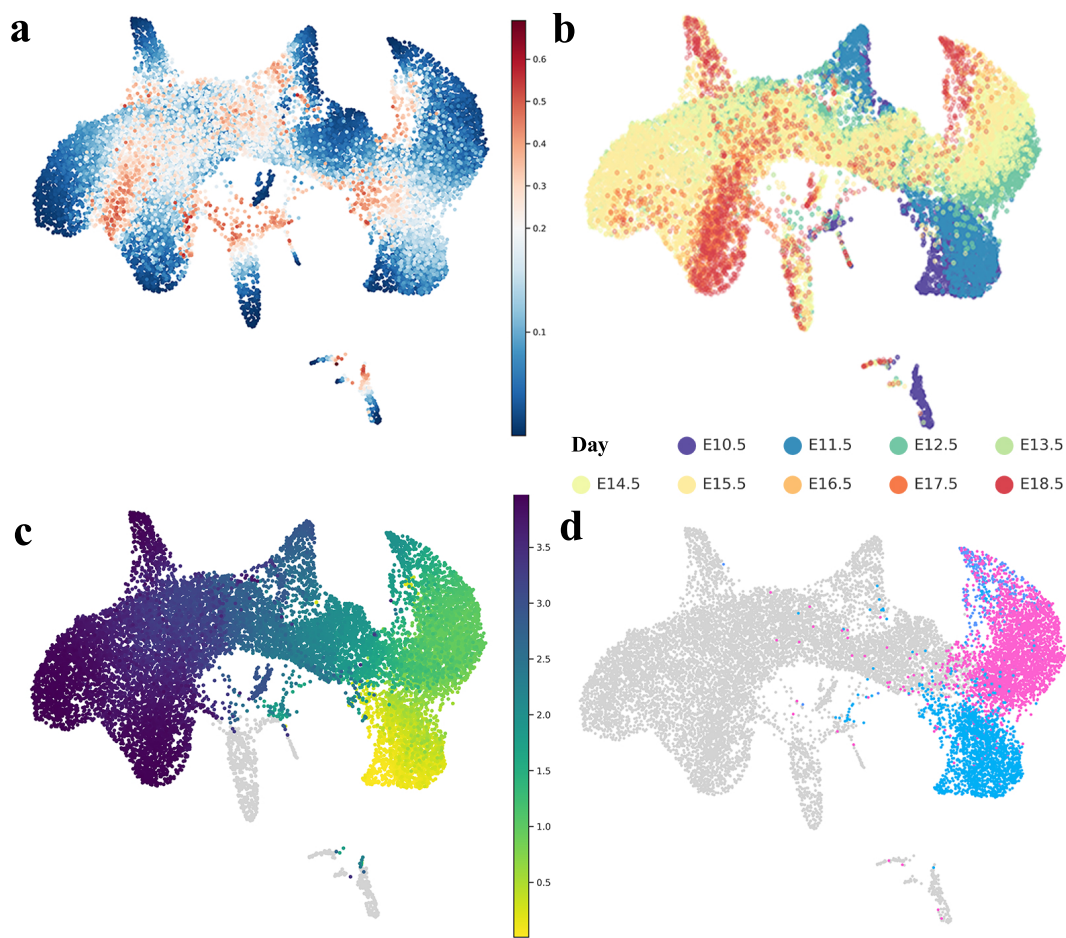
Fig. S1. Six different topologies of the underlying trajectories in real and synthetic datasets.



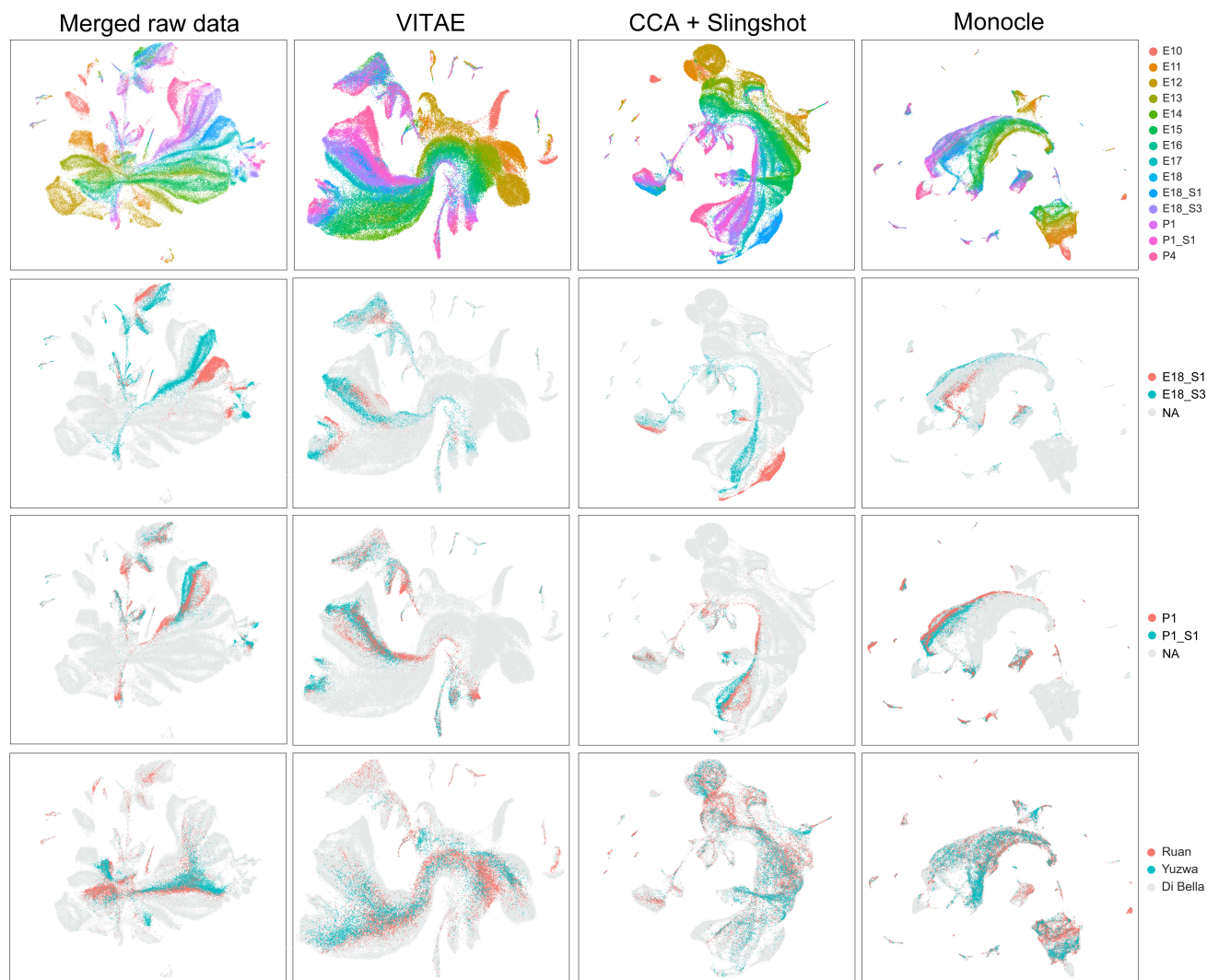
**Fig. S2.** Differential expressed testing results on simulated datasets generated by dyntoy. The false discovery ratio (FDR) and true positive ratio (TPR) are shown in the left and right panels, respectively.



**Fig. S3. Comparison of stability with/without the Jacobian regularization.** VITAE is applied to perform a joint analysis of Yuzwa and Ruan datasets with 10 repeated trials with random initialization. Top left: standard derivations across 10 trials for the pairwise distances of the estimated cluster centers  $U$  in the latent space. Top right: standard derivations across 10 trials for the estimated probabilities being assigned to each edge and vertex ( $\pi$ ). Bottom left: standard derivations across 10 trials for the evaluation metric values between each trial's final result and the illustrated trajectory inference result in Figure 3. Bottom right: mean across 10 trials for the evaluation metric values between each trial's final result and the illustrated trajectory inference result in Figure 3.



**Fig. S4. UMAP visualizations of VITAE's low-dimensional cell representations on the joint analysis of Yuzwa and Ruan datasets.** (a) Low-dimensional embeddings colored by the projection uncertainty estimated by VITAE. When projected to the inferred trajectory, Cells colored in red have higher uncertainties compared to cells colored in blue. (b) Low-dimensional embeddings colored by collection days. (c) Low-dimensional embeddings colored by the pseudotime estimated by VITAE. Cells not in the inferred trajectory are colored in gray. (d) Low-dimensional embeddings highlighting the NEC-RGC-OPC sub-trajectory where the cell types NEC, RGC, and OPC are highlighted.



**Fig. S5. UMAP visualizations of the joint analysis of Di Bella, Yuzwa and Ruan datasets.** Each column represents a method. For the first column, cells are normalized and then concatenated to perform a joint PCA analysis before UMAP. In the first row and last row, cells are colored by their collection days and sources. In the second and third rows, cells from the two replicates at E18 and two replicates at P1 from the Di Bella dataset are highlighted.

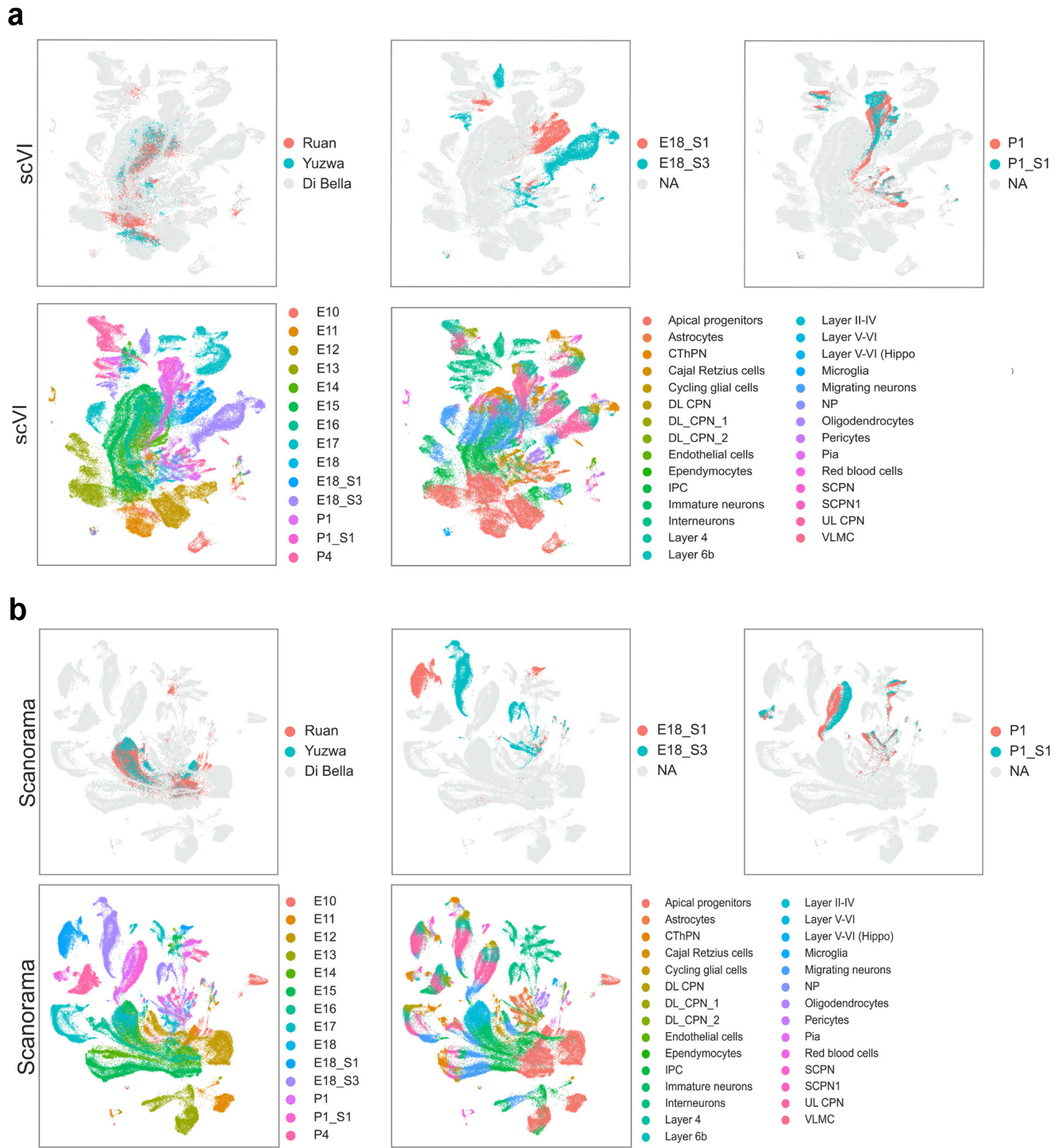
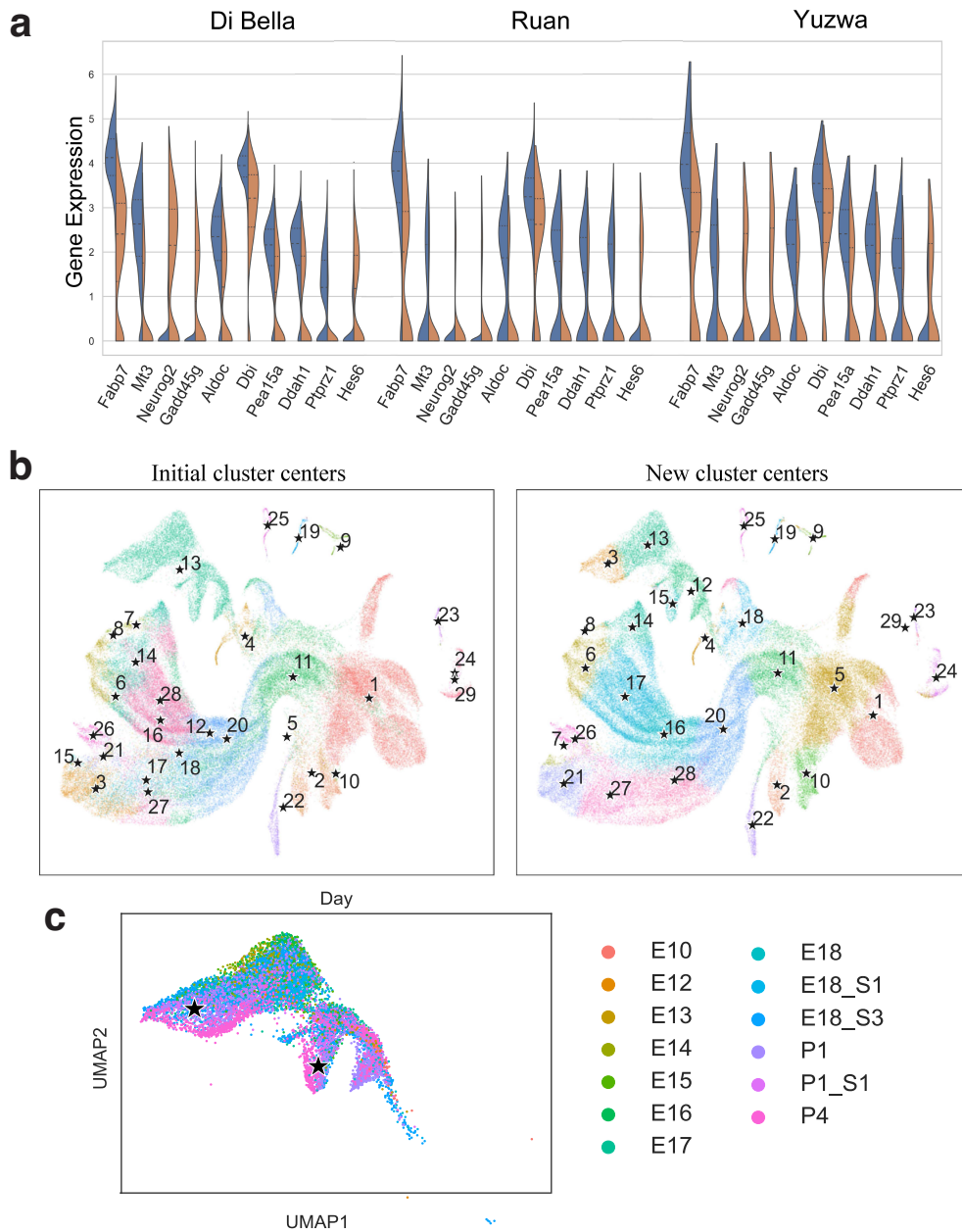
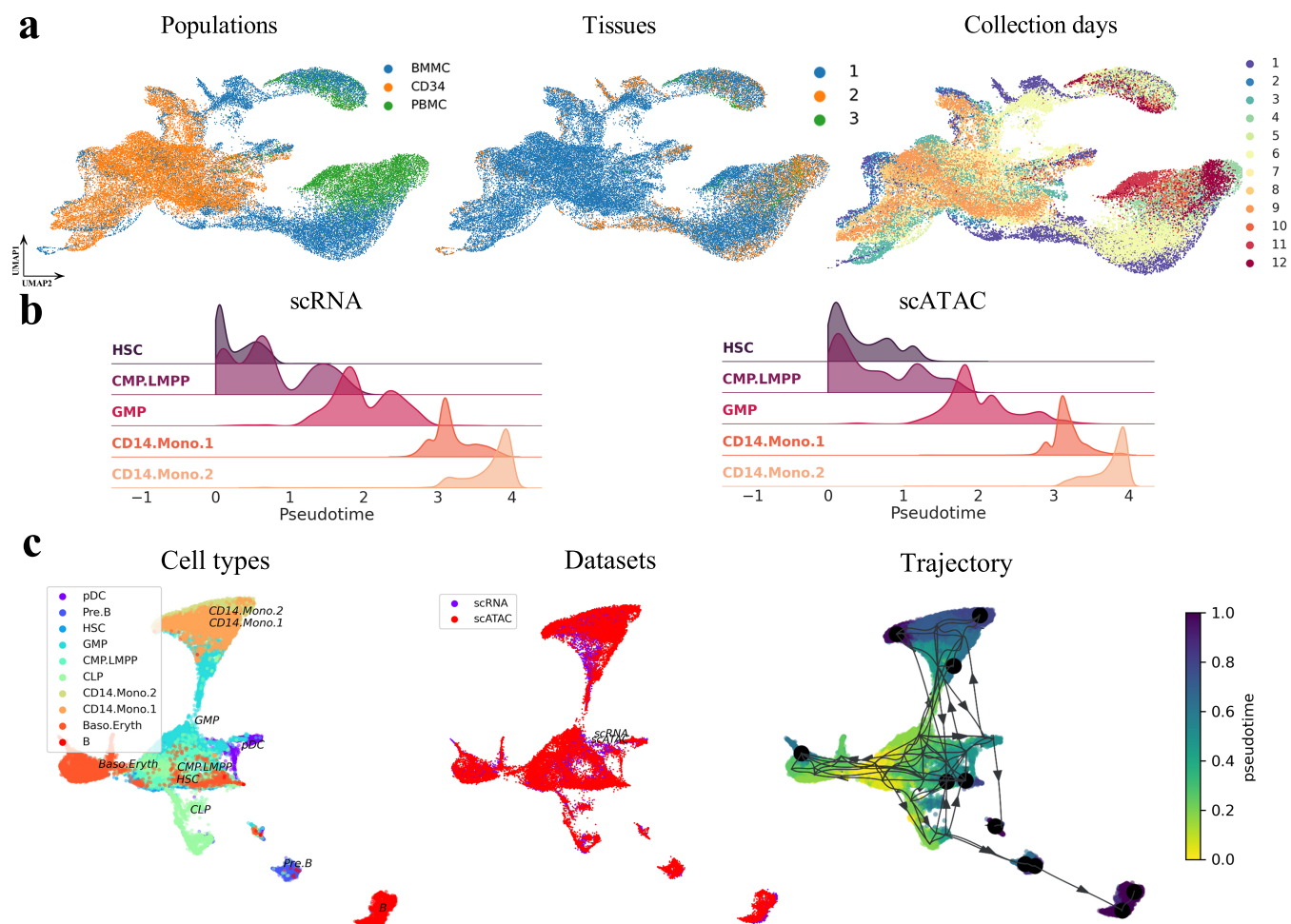


Fig. S6. UMAP visualizations of the integration of Di Bella, Yuzwa and Ruan datasets using scVI and Scanorama.



**Fig. S7. Additional results for joint trajectory analysis of Di Bella, Yuzwa and Ruan datasets.** (a) The violin plots of the top 10 differentially expressed gene expressions between the APs in the glial and neuronal branches in Fig. 4c. Tests are done by combining all three datasets while gene expressions are visualized for each dataset separately. (b) Locations of the cluster centers before and after VITAE training for the joint analysis of Di Bella, Yuzwa, and Ruan datasets. VITAE uses the reference cell types to initialize the locations of the vertices before training, and then the vertices will change locations and give new clustering results after training. The left and the right panels show the initial and trained cluster centers, respectively. For interneurons, though only one vertex (cluster 13) is initialized, VITAE can automatically learn four cluster centers after training. (c) Interneurons colored by collection days.



**Fig. S8. Integrative trajectory inference of multi-omic human hematopoiesis data.** (a) UMAP visualization of VITAE's low-dimensional embedding of cells, colored by cell populations ('BMMC', 'CD34', and 'PBMC' stand for bone marrow mononuclear cells, CD34<sup>+</sup>-enriched bone marrow mononuclear cells, and peripheral blood mononuclear cells, respectively), tissues, replicates, and collection days. VITAE merges the gene expressions (scRNA) and the gene activities (scATAC) score, retains meaningful biological variations and correctly infers the developmental trajectory. (b) Distribution of pseudotime of the monocytic lineage for scRNA-seq and scATAC-seq cells. (c) Trajectory inference results of VIA (36).

631 **References**

- 632 1. W Saelens, R Cannoodt, H Todorov, Y Saeys, A comparison of single-cell trajectory inference methods. *Nat. biotechnology*  
633 **37**, 547–554 (2019).
- 634 2. C Trapnell, et al., The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single  
635 cells. *Nat. biotechnology* **32**, 381 (2014).
- 636 3. D Agarwal, J Wang, NR Zhang, Data denoising and post-denoising corrections in single cell RNA sequencing. *Statistical*  
637 *Science* **35**, 112–128 (2020).
- 638 4. G Eraslan, LM Simon, M Mircea, NS Mueller, FJ Theis, Single-cell RNA-seq denoising using a deep count autoencoder.  
639 *Nat. communications* **10**, 1–14 (2019).
- 640 5. DP Kingma, M Welling, Auto-encoding variational bayes in *2nd International Conference on Learning Representations*,  
641 eds. Y Bengio, Y LeCun. (2014).
- 642 6. K Sohn, H Lee, X Yan, Learning structured output representation using deep conditional generative models in *Advances*  
643 *in neural information processing systems*. pp. 3483–3491 (2015).
- 644 7. I Higgins, et al., beta-vae: Learning basic visual concepts with a constrained variational framework in *International*  
645 *Conference on Learning Representations*. (2016).
- 646 8. M Huang, Z Zhang, NR Zhang, Dimension reduction and denoising of single-cell RNA sequencing data in the presence of  
647 observed confounding variables (2020).
- 648 9. DP Kingma, J Ba, Adam: A method for stochastic optimization. *Int. Conf. on Learn. Represent.* **0** (2015).
- 649 10. DJ Rezende, S Mohamed, D Wierstra, Stochastic backpropagation and approximate inference in deep generative models  
650 in *International Conference on Machine Learning*. pp. 1278–1286 (2014).
- 651 11. A Stuhlmüller, J Taylor, N Goodman, Learning stochastic inverses. *Adv. neural information processing systems* **26**,  
652 3048–3056 (2013).
- 653 12. S Frühwirth-Schnatter, G Malsiner-Walli, From here to infinity: sparse finite versus dirichlet process mixtures in  
654 model-based clustering. *Adv. data analysis classification* **13**, 33–64 (2019).
- 655 13. TN Tran, GD Bader, Tempora: Cell trajectory inference using time-series single-cell RNA sequencing data. *PLoS*  
656 *computational biology* **16**, e1008205 (2020).
- 657 14. J Wang, Q Zhao, T Hastie, AB Owen, Confounder adjustment in multiple hypothesis testing. *Annals statistics* **45**, 1863  
658 (2017).
- 659 15. K Van den Berge, et al., Trajectory-based differential expression analysis for single-cell sequencing data. *Nat. communica-*  
660 *tions* **11**, 1–13 (2020).
- 661 16. D Song, JJ Li, Pseudotime: inference of differential gene expression along cell pseudotime with well-calibrated p-values  
662 from single-cell rna sequencing data. *Genome biology* **22**, 124 (2021).
- 663 17. FA Wolf, P Angerer, FJ Theis, Scanpy: large-scale single-cell gene expression data analysis. *Genome biology* **19**, 1–5  
664 (2018).
- 665 18. VD Blondel, JL Guillaume, R Lambiotte, E Lefebvre, Fast unfolding of communities in large networks. *J. statistical*  
666 *mechanics: theory experiment* **2008**, P10008 (2008).
- 667 19. AL Maas, AY Hannun, AY Ng, Rectifier nonlinearities improve neural network acoustic models in *International conference*  
668 *on machine learning*. Vol. 30, p. 3 (2013).
- 669 20. S Ioffe, C Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift in  
670 *International conference on machine learning*. (pmlr), pp. 448–456 (2015).
- 671 21. S Rifai, P Vincent, X Muller, X Glorot, Y Bengio, Contractive auto-encoders: Explicit invariance during feature extraction  
672 in *Proceedings of the 28th international conference on international conference on machine learning*. pp. 833–840 (2011).
- 673 22. D Jakubovitz, R Giryes, Improving dnn robustness to adversarial attacks using jacobian regularization in *Proceedings of*  
674 *the European Conference on Computer Vision (ECCV)*. pp. 514–529 (2018).
- 675 23. M Amodio, et al., Exploring single-cell data with deep multitasking neural networks. *Nat. methods* **16**, 1139–1145 (2019).
- 676 24. M Lotfollahi, et al., Mapping single-cell data to reference atlases by transfer learning. *Nat. Biotechnol.* **40**, 121–130 (2022).
- 677 25. M Lotfollahi, M Naghipourfar, FJ Theis, FA Wolf, Conditional out-of-distribution generation for unpaired data using  
678 transfer vae. *Bioinformatics* **36**, i610–i617 (2020).
- 679 26. GX Zheng, et al., Massively parallel digital transcriptional profiling of single cells. *Nat. communications* **8**, 1–12 (2017).
- 680 27. R Cannoodt, W Saelens, L Deconinck, Y Saeys, dyngen: a multi-modal simulator for spearheading new single-cell omics  
681 analyses (2020).
- 682 28. Z Abu-Aisheh, R Raveaux, JY Ramel, P Martineau, An exact graph edit distance algorithm for solving pattern recognition  
683 problems in *4th International Conference on Pattern Recognition Applications and Methods*. (2015).
- 684 29. G Jurman, R Visintainer, M Filosi, S Riccadonna, C Furlanello, The HIM glocal metric and kernel for network comparison  
685 and classification in *2015 IEEE International Conference on Data Science and Advanced Analytics*. (IEEE), pp. 1–10  
686 (2015).
- 687 30. L Hubert, P Arabie, Comparing partitions. *J. classification* **2**, 193–218 (1985).
- 688 31. T Stuart, et al., Comprehensive integration of single-cell data. *Cell* **177**, 1888–1902 (2019).
- 689 32. J Cao, et al., The single-cell transcriptional landscape of mammalian organogenesis. *Nature* **566**, 496–502 (2019).
- 690 33. A Gayoso, et al., A python library for probabilistic analysis of single-cell omics data. *Nat. biotechnology* **40**, 163–166  
691 (2022).

- 692 34. B Hie, B Bryson, B Berger, Efficient integration of heterogeneous single-cell transcriptomes using scanorama. *Nat.*  
693 *biotechnology* **37**, 685–691 (2019).
- 694 35. MD Luecken, et al., Benchmarking atlas-level data integration in single-cell genomics. *Nat. methods* **19**, 41–50 (2022).
- 695 36. SV Stassen, GG Yip, KK Wong, JW Ho, KK Tsia, Generalized and scalable trajectory inference in single-cell omics data  
696 with via. *Nat. communications* **12**, 5528 (2021).
- 697 37. SA Yuzwa, et al., Developmental emergence of adult neural stem cells as revealed by single-cell transcriptional profiling.  
698 *Cell reports* **21**, 3970–3986 (2017).
- 699 38. X Ruan, et al., Progenitor cell diversity in the developing mouse neocortex. *Proc. Natl. Acad. Sci.* **118**, e2018866118  
700 (2021).
- 701 39. DJ Di Bella, et al., Molecular logic of cellular diversification in the mouse cerebral cortex. *Nature* **595**, 554–559 (2021).
- 702 40. MS Kowalczyk, et al., Single-cell RNA-seq reveals changes in cell cycle and differentiation programs upon aging of  
703 hematopoietic stem cells. *Genome research* **25**, 1860–1872 (2015).
- 704 41. S Petropoulos, et al., Single-cell RNA-seq reveals lineage and X chromosome dynamics in human preimplantation embryos.  
705 *Cell* **165**, 1012–1026 (2016).
- 706 42. F Guo, et al., The transcriptome and DNA methylome landscapes of human primordial germ cells. *Cell* **161**, 1437–1452  
707 (2015).
- 708 43. KM Loh, et al., Mapping the pairwise choices leading from pluripotency to human bone, heart, and other mesoderm cell  
709 types. *Cell* **166**, 451–467 (2016).
- 710 44. H Hochgerner, A Zeisel, P Lönnerberg, S Linnarsson, Conserved properties of dentate gyrus neurogenesis across postnatal  
711 development revealed by single-cell rna sequencing. *Nat. neuroscience* **21**, 290–299 (2018).
- 712 45. B Treutlein, et al., Dissecting direct reprogramming from fibroblast to neuron using single-cell RNA-seq. *Nature* **534**,  
713 391–395 (2016).
- 714 46. FA Wolf, et al., PAGA: graph abstraction reconciles clustering with trajectory inference through a topology preserving  
715 map of single cells. *Genome biology* **20**, 1–9 (2019).